A MULTI-LEVEL PREDICTIVE METHODOLOGY FOR TERMINAL AREA AIR TRAFFIC FLOW

A Dissertation
Presented to
The Academic Faculty

By

Sun Choi

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology

May 2019

A MULTI-LEVEL PREDICTIVE METHODOLOGY FOR TERMINAL AREA AIR TRAFFIC FLOW

ACKNOWLEDGEMENTS

First of all, I would like to express my deep appreciation to my advisor, Professor Dimitri Mavris, for his guidance through the PhD program. Dr Mavris supported me to complete my PhD successfully and provided generous advice for a better career. Without his guidance and persistent help, I would have had difficulty finishing my PhD.

I am deeply grateful to my thesis committee members: Dr Simon Briceno, Professor Daniel Schrage, Professor Haomin Zhou, and Dr Vikram Saletore. I thank them for serving on my committee members and taking their time to review my dissertation. I especially would like to thank Dr Briceno for giving valuable advice and techinical guidance since I worked with him for projects at Aerospace Systems Design Laboratory. In addition, I acknowledge Dr Saletore who has helped me as a committee member as well as a manager.

I would like to thank all my friends and colleagues at Aerospace Systems Design Laboratory who helped me along the way. I especially thank Dr Youngchul Park, Dr Woongje Sung, and Eric Inclan for their constructive criticism on my research and for having an enjoyable time together. I thank Dr Youngjoon Choi for helping me prepare for the qualifying exam. I also thank Adrienne Durham for coordinating forms about graduation and making my life easier. I thank Glenn Campopiano for handling financial affairs and always giving kind words. Furthermore, I would like to thank Dr Young Jin Kim for not only introducing me to the topic of my thesis but also giving me generous advice after that. In addition to my colleagues, I sincerely thank Dr Soonyoung Cha, who gave me great support and made my life a pleasant one.

Finally, I extend my sincere gratitude to my parents and my brother for their unconditional love and support throughout the entire process.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Over the past few decades, the air transportation system has grown significantly. In particular, the number of passengers using air transportation has greatly increased. As the demand for air travel expands, airport departure/arrival demand almost reaches its capacity. In consequence, the level of delays increases since the system capacity cannot manage the increased demand. With this trend, the national airspace system (NAS) will be saturated, and the congestion at the airport will become even more severe.

As a result of congestion, a considerable number of flights experience delays. According to the Bureau of Transportation Statistics (BTS), over 1 million flights are operated in a year, and about twenty percent of all scheduled commercial flights are delayed more than 15 minutes. These delays cost billions of dollars annually for airlines, passengers, and the US economy. Therefore, this study seeks to find out why the delays occur and to analyze patterns in which the delays occurred.

Analysis of airport operations generally falls into a macro or micro perspective. At the macro point of view, very few details are considered, and delays are aggregated at the airport level. Especially, shortfalls in airport capacity and a capacity-demand imbalance are the primary causes of delays in this respect. In the micro perspective, each aircraft is modeled individually, and the causes of delays are reproduced as precisely as possible. Micro reasons for air traffic delays include inclement weather, mechanics problems, operation issues. In this regard, this research proposes a methodology that can efficiently and practically predict macro and micro-level air traffic flow in the terminal area.

For a macro-level analysis of delays, artificial neural networks models are proposed to predict the hourly airport capacity. Multi-layer perceptron (MLP), recurrent neural network (RNN), and long short-term memory (LSTM) are trained with historical weather and airport capacity data of Hartsfield-Jackson Atlanta airport (ATL). In the performance evaluation, the models have presented decent predictive performance and successfully predicted the test data as well as the training data.

On the other hand, Random Forests and AdaBoost are implemented in the micro-level modeling of the air traffic. The micro-level models trained with on-time flight performance data and corresponding weather data focus on a classification of the individual flight delays. The model provides interpretability and imbalanced data handling while the accuracy is as good as the existing methods. Lastly, the predictive model for individual flight delays is refined using the cost-proportionate rejection sampling (costing) method. Along with the integration of the costing method, general machine learning algorithms have been converted to cost-sensitive classifiers. The cost-sensitive classifiers were able to account for asymmetric misclassification costs without losing their diagnostic functionality as binary classifiers.

This study presents a data-driven approach to air traffic flow management that can effectively utilize air traffic data accumulated over decades. Through data analysis from the macro and micro perspective, an integrated methodology for terminal air traffic flow prediction is provided. An accurate prediction of the airport capacity and individual flight delays will assist stakeholders in taking more informed decisions.

CHAPTER 1
MOTIVATION

1.1 Growing Aviation Activity

Air transportation is a more complex distributed system than any other transportation system, with the task of handling sophisticated schedules [1]. The reason is that air transportation services must meet the diverse demands of passengers under limited resources and at the same time be safe and reliable. Also, the increasing number of passengers has made the air transportation system more complicated. The number of passengers using air transportation has dramatically increased over the past few decades. According to Figure 1.1, the number of passengers has tripled over the decades. After the September 11 attacks, the demand for air traffic dropped sharply, but it has recovered its upward trend soon. By 2018, the number of passengers enplaned is expected to become three times as much as that of 1978. This increasing trend is indeed a tremendous development in the air transportation industry. More than 800 millions people use air transportation per year, which means that airlines provide services to 2 million passengers every day.

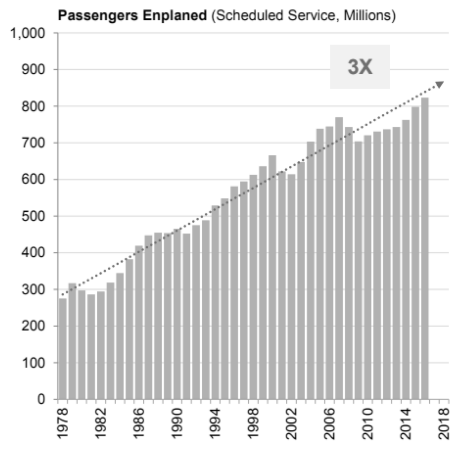


Figure 1.1: The Number of Passengers Enplaned [2]

The development of air transportation is expected to continue over the next few years. The Federal Aviation Administration (FAA) predicts air traffic demand in the United States based primarily on changes in income and population, and these variables are expected to increase [3]. A revenue passenger miles (RPM) is a good indicator reflecting passenger demand defined by the Bureau of Transportation Statistics (BTS). In Figure 1.2, the Federal Aviation Administration's forecast of domestic RPM is presented. The baseline, pessimistic and optimistic forecasts are based on assumptions about the economic growth over the next decades. In the optimistic scenario, 3.2% annual growth in revenue passenger miles is expected. Moreover, it is forecasted to grow 2% per year even for the pessimistic scenario. At current trends, RPM values after 20 years will show 40% growth even in the pessimistic forecasts [4]. According to the statistics, there is no doubt that the passenger demand will increase. Problems can arise when the air traffic system fails to respond appropriately to the rapidly growing demand [5].

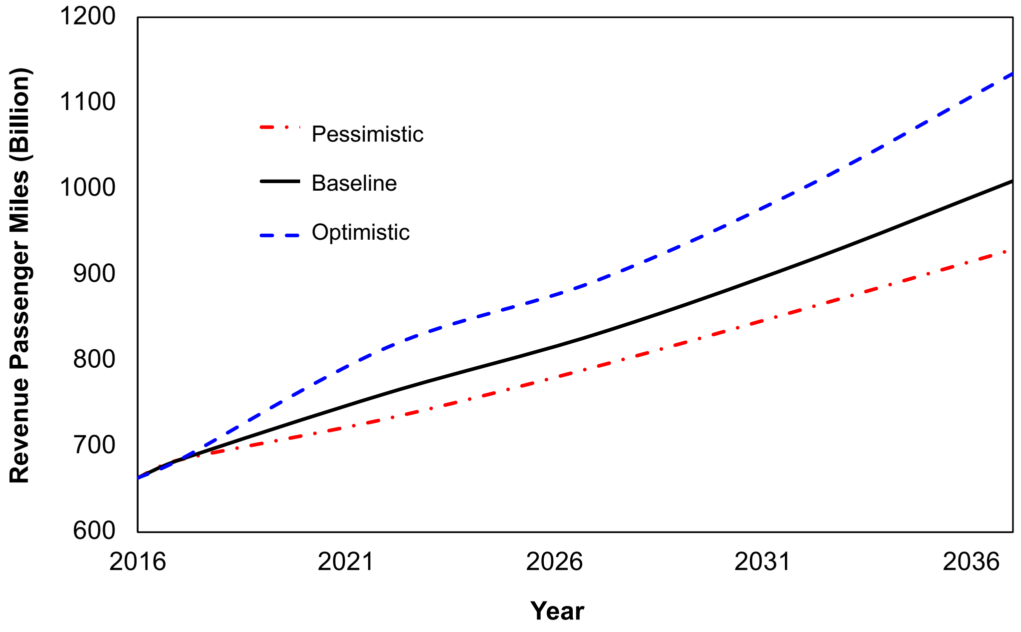


Figure 1.2: Forecast of Domestic Aviation Activity [4]

1.2 Expected Congestion in the NAS

The rate of increase in passenger demand for air transportation is faster than the capacity improvement of the National Airspace System (NAS) in the United States [6]. As predicted in Section 1.1, the demand for air traffic will increase further over the next 20 years, and the NAS will be saturated quickly if the capacity does not expand. Consequently, the congestion at airports will become even more severe with the current trend. Expected level of congestion in the NAS without further improvements beyond near-term NextGen is shown in Figure 1.3. Figure 1.3 shows that the NAS is quite saturated. The red bar indicates the percentage of hours that there is a high level of delays in departure or arrival between 7 am to 11 pm. For example, Hartsfield-Jackson Atlanta (ATL) will experience severe delays during 89% of the day's operating hours in 2030. According to Future Airport Capacity Task (FACT) 3 reports, 12 airports among 30 core airports in the NAS will be severely capacity-constrained without further improvement beyond near-term NextGen. That is why the FAA plans to improve throughput by increasing the number of runways at some airport,

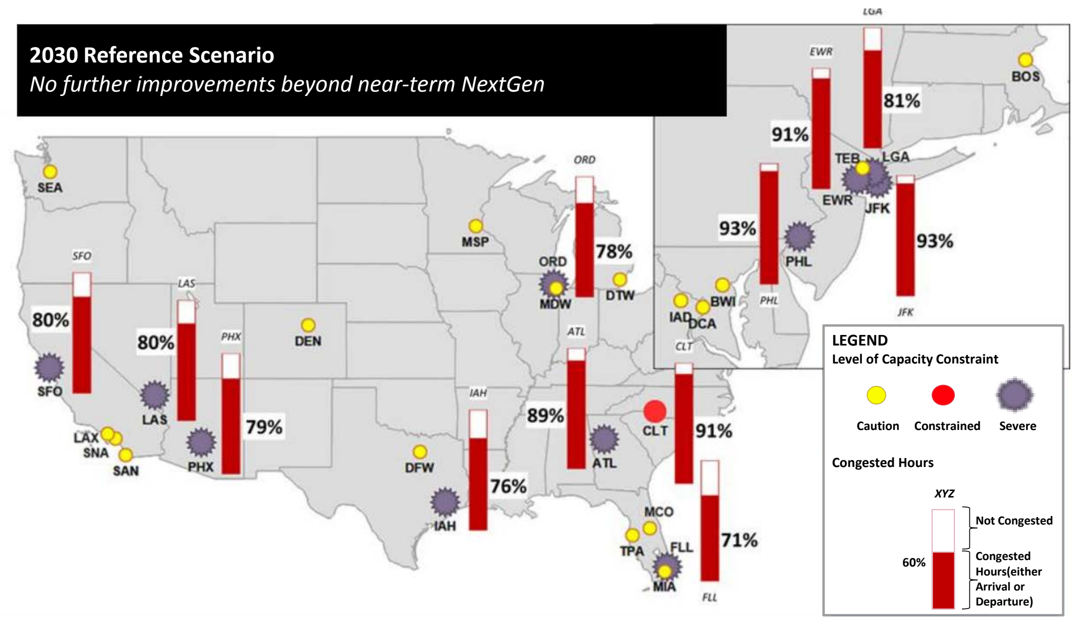


Figure 1.3: Forecasted Airport Congestion in 2030, Assuming No Future Improvement beyond Near-Term NextGen [7]

or by NAS modernization. However, despite the diverse efforts to lower the congestion level, congestion at the airport is inevitable as demand increases.

1.3 Air Traffic Flow Management

The Federal Aviation Administration (FAA) strives to prevent congestion in the US air traffic systems in response to air traffic demand increase. As part of this effort, the FAA has implemented Air Traffic Flow Management (ATFM). ATFM is a process that balances the user demand with airspace capacity for safety and efficiency of the NAS [8]. When the system capacity is larger than the demand, it is relatively easy to achieve a safe, orderly and expeditious operation without ATFM. When the demand exceeds the capacity, ATFM is necessary to have efficient and effective air transportation system as shown in Table 1.1. In a capacity-constrained system like the current NAS, an orderly and expeditious operation is not guaranteed without ATFM.

Table 1.1: Benefit of Air Traffic Flow Management (ATFM) [9]

		Safe	Orderly	Expeditious
Capacity $\geq$ Demand		○	○	○
Capacity $<$ Demand	With ATFM	△	$\times$	$\times$
	With ATFM	○	○	○

There has been a series of effort belonging to ATFM. Examples of ATFM are Ground Delay Programs, Continuous Descent Approaches (CDA) and en-route sector capacity management. A Ground Delay Program (GDP) is a traffic management procedure that delays the departure of an aircraft at the origin airport to manage the demand and capacity of the destination airport [10]. When the demand is expected to exceed the capacity of the destination airport due to weather deterioration, runway closure, etc., the GDP reduces the number of incoming flights into the airport per hour. The motivation for the GDP is that delays are inevitable and it is safer and cheaper to have delays on the ground rather than en-route [11].

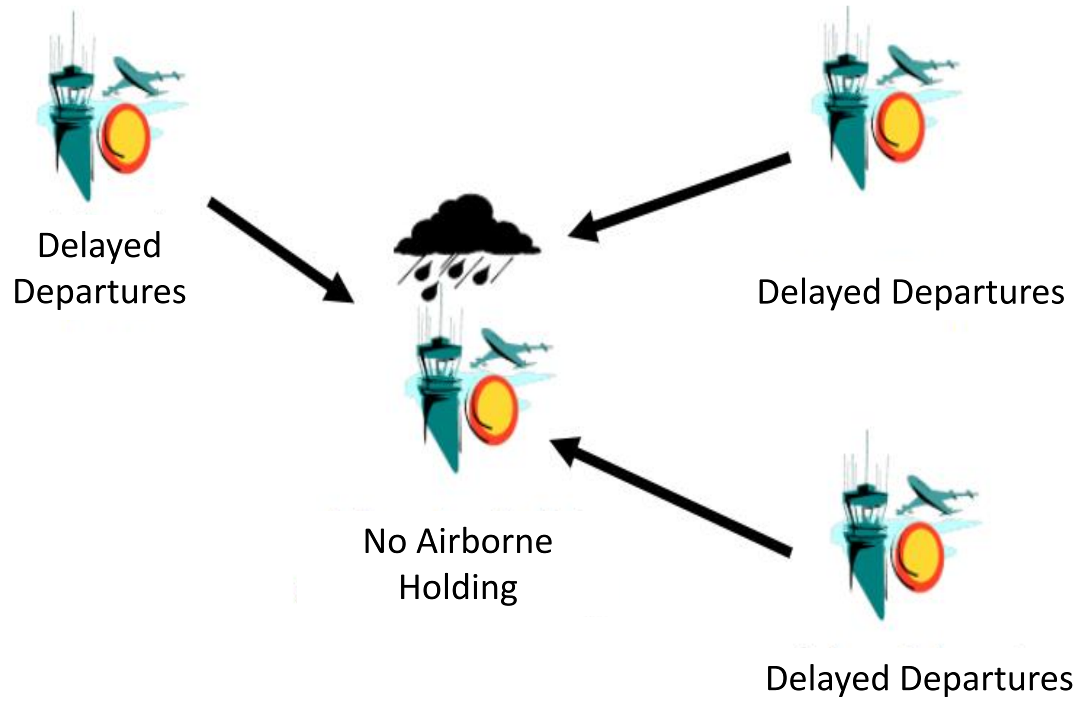


Figure 1.4: Ground Delay Programs Delaying Flights to the Airport when Congestion is Expected at the Airport [12]

Since 1998, the FAA has introduced a new procedure to make GDP decisions, called Collaborative Decision Making (CDM). CDM is a collaborative effort between the FAA, airlines and airports. It aims to improve the traffic flow management system through information exchange, data sharing and improved automated decision support tools. Members participating in CDM continue to send the FAA operational schedules and changes such as delays, cancellations, and newly created flights [13]. The exchange of information between participants provides an accurate estimate of the projected demand and benefits all users by making better decisions regarding GDP.

The application of Continuous Descent Approaches (CDA) has been discussed to develop an advanced air traffic management procedures to accommodate demand increase in the congested terminal area [14] [15] [16]. A conventional landing approach is a procedure in which an aircraft repeatedly requests permission to descend to a lower altitude and approaches the airport in a stepwise manner. CDA is a procedure that contrasts with a conventional landing approach which is evident in Figure **??**. It refers to the process by which an aircraft descends continuously from cruise to landing [14]. Airlines can achieve potential savings from CDA by reducing flight time, fuel consumption and noise [16]. However, CDA requires the extra spacing to ensure separation, which can reduce arrival throughput and increase delays [17].

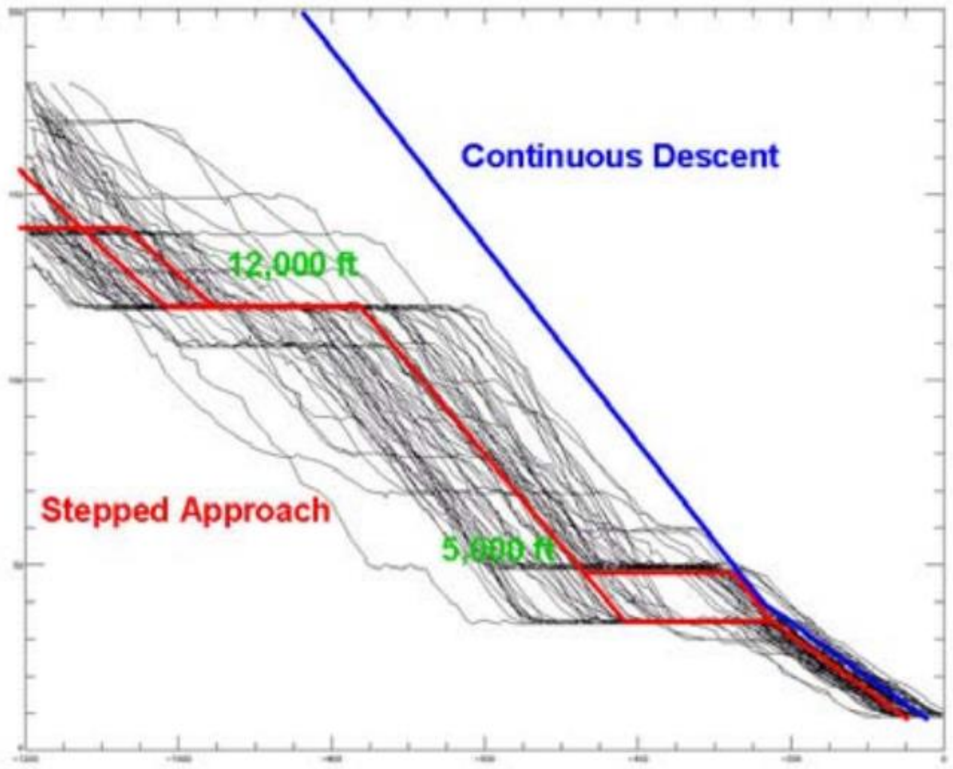


Figure 1.5: Descent Profiles of Continuous Descent Approaches and Stepwise Descent to ATL [16]

Lastly, there is an approach to managing air traffic concerning en-route sector capacity. En-route sector congestion is the primary factor of flight delays in Europe [18]. This is a secondary factor in the NAS delays, but it is still important. For this reason, researches on the capacity of an air traffic control (ATC) sector have been performed [19] [5] [20]. Air traffic control of specific sub-areas within the NAS is delegated to the jurisdiction of one of the 22 air route traffic control centers (ARTCCs) [10]. ARTCCs in the NAS are in Figure 1.6 except for Anchorage (ZAN) and Honolulu (ZHN). ARTCC airspace is split into smaller parts called ATC sectors. ATC is responsible for the safe and efficient management of air traffic flow through the sector [19]. Sector capacity is the number of flights that an ATC can handle within a sector for a specified time. In general, the human workload is a critical factor in determining ATC capacity, as there is a limit on the amount that a person can handle per given hour [20]. Therefore, a considerable number of studies modeled and quantified the ATC workload and the sector capacity together [21].

1.4 Air Traffic Delays

Because flight delays are a symptom of the inefficiency in air traffic systems [22], a general performance indicator for air traffic operations is punctuality within 15 minutes [23]. Airline delays occur when demand reaches the maximum capacity of the air traffic network. As the demand for air travel grows, the airport departure/arrival capacity becomes saturated, and flights are delayed.

Figure 1.7 shows the number of flights operated in a year and the percent of delayed flights from the statistics of the Bureau of Transportation Statistics (BTS). The line chart represents the total number of flights operated in a year and the bar chart shows the percentage of delayed arrivals operated by major carriers. More than 1 million flights are operated in a year, and approximately twenty percent of the scheduled commercial flights are delayed more than 15 minutes. In 2016 alone, 240000 flights from major carriers were delayed.

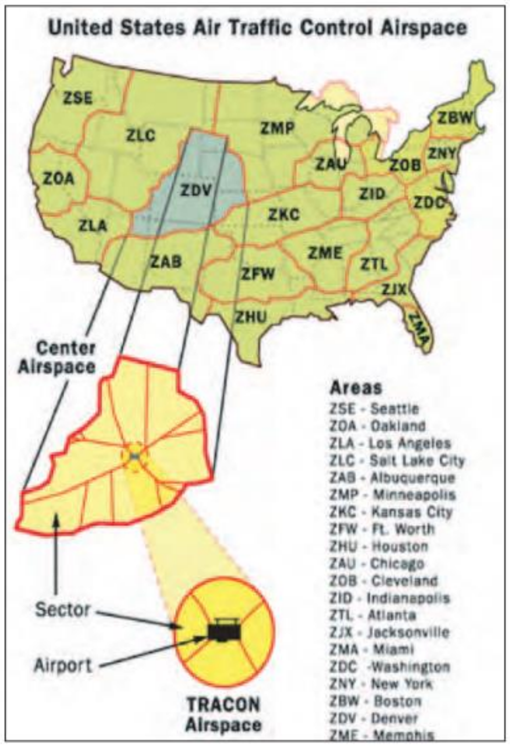


Figure 1.6: FAA Air Route Traffic Control Centers except for Anchorage (ZAN) and Honolulu (ZHN) [10]

Delays are a huge problem because they cost multi-billion dollars annually for airlines, passengers and the US economy [25]. As a concrete example, the cost of delays for 2016 is detailed in Figure 1.8. First of all, the biggest part, which accounts for more than half of the total cost, is the passenger's value of time. Passengers will waste additional buffer time when they miss the connecting flight due to delays. In the case of direct flights, passengers can miss important meetings or appointments as a consequence of delays. After the passengers, airlines suffer the next significant loss. The loss is due to increased crew, fuel and maintenance costs. The third column in Figure 1.8, 'Lost demand', means the expected loss from passengers avoiding air transportation due to delays. Item marked as indirect includes increased cost due to a loss of productivity in the relevant business [2].

The problems caused by delays are not limited to cost.One of the biggest reasons that passengers complain is delays. When delays occur, passengers have to wait from tens of

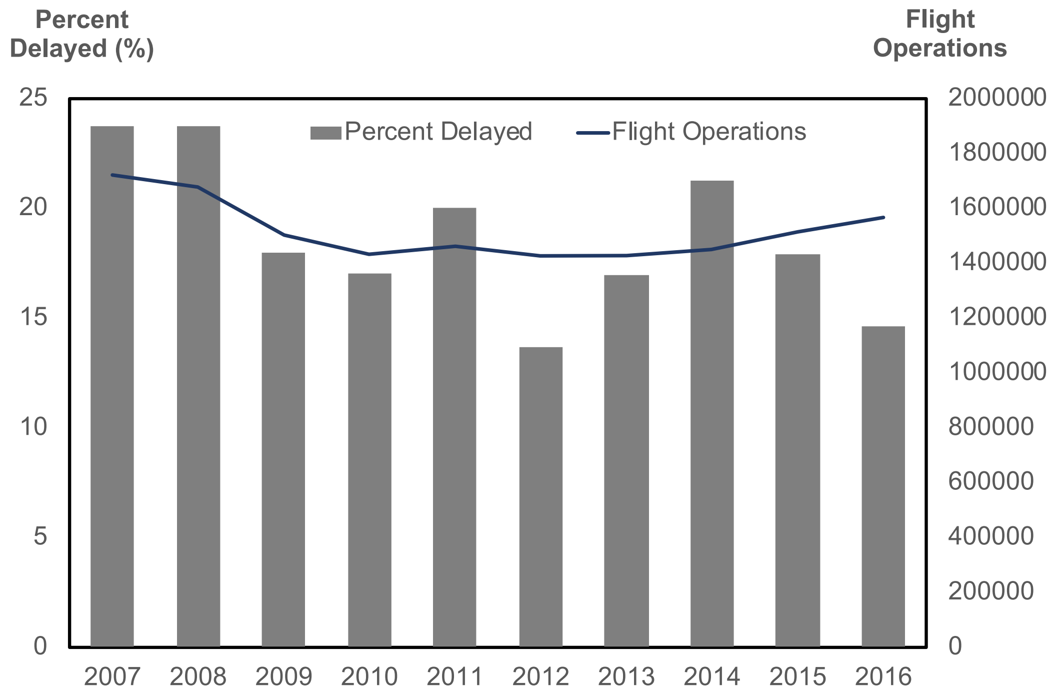


Figure 1.7: The Number of Flights Operated and Percent Delayed [24]

minutes to hours in the uncomfortable chair at the airport. Passengers concerned about the uncertainty of the delay occurrence may pay extra to arrive early enough [1]. Performance reliability can affect customer loyalty and jeopardize airlines marketing strategies [1]. From an environmental point of view, delays in congested airports result in unnecessary fuel consumption and CO_2 emission [26] [27]. For ground delays, additional emissions can be caused by auxiliary power units (APUs) [27]. On the other hand, airborne holding will consume more fuel by flying longer trajectory than the initial plan [28].

As listed above, delays cause enormous economic losses, the passenger inconvenience and environmental pollution, and they motivate the development of delay analysis and management mechanisms [6]. If flight delays can be predicted by exploring the causes, it will be able to contribute to reducing the losses of the stakeholders. Airlines will be able to

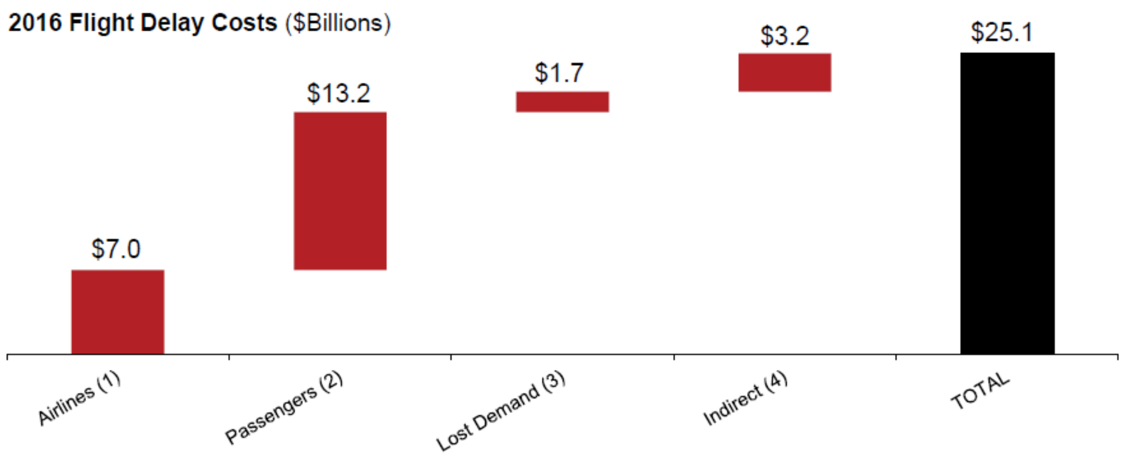


Figure 1.8: Details of Costs due to Flight Delays in 2016 [2]

secure customer loyalty by providing accurate information to passengers. With this information, passengers will be able to adjust their plans in advance to minimize their losses. For these reasons, this study looks at the causes of delays and how they can be used to predict delays.

1.5 Causes of Flight Delays

1.5.1 Categorization according to Flight Phase

A commercial flight begins by someone boarding an aircraft with the intent of flying. The flight is terminated when all persons who boarded with the intention of flying get off [29]. From the beginning to the end of the flight, delays can occur from a variety of sources as the aircraft undergoes numerous stages in the airport, runway and airspace. Figure 1.9 illustrates the conventional steps of commercial flights. In the following, the factors that contribute to the delays for each step will be described.

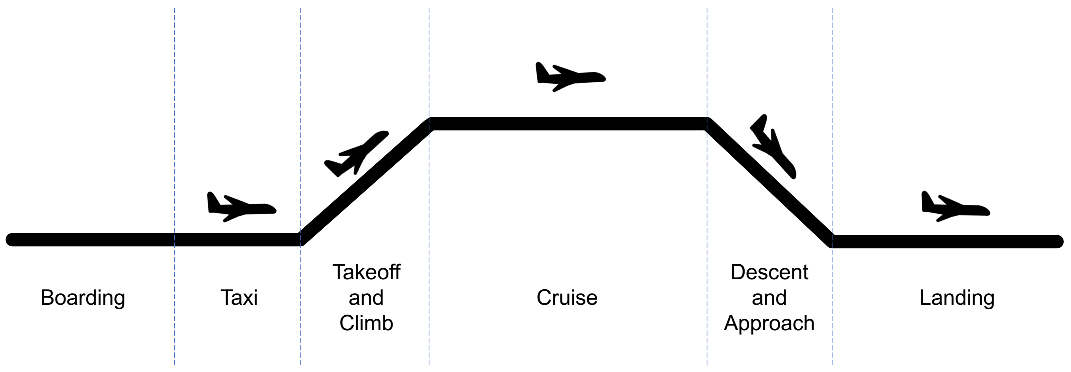


Figure 1.9: Conventional Phases of a Commercial Flight

One of the causes of delays during the boarding phase is a pilot or a crew member. When the schedule of the pilot is intertwined or their boarding is delayed for some reasons, but the airline can not assign an alternative pilot, passengers must wait until the pilot arrives. On the other hand, flights sometimes wait for passengers trapped in long lines of security. Other reasons may be mechanical defects in the aircraft. The FAA requires strict mechanical restrictions on the aircraft and the aircraft cannot depart unless it satisfies them. Regulatory requirements include rigorous criteria for operational status including system redundancy as well as periodic checks. Repairs and maintenance can take from a few minutes to several hours.

The next step in boarding is a taxi. A taxi is a phase in which an aircraft moves on the ground with its power after all passengers and crew members have boarded it [29]. The delay that occurs during a taxi is called a tarmac delay. This is the case when the aircraft leaves the gate and awaits takeoff from the runway, or is waiting on the runway after landing. A tarmac delay often occurs when the runway is particularly busy. Flights that are in the taxi stage have to wait for priority flights to use the runway first because the priority of the aircraft that are landing or taking off is higher [30]. As there is no opportunity for passengers to get off the plane while tarmac delays occur [31], the US Department of Transportation regulations require airlines to provide snacks, adequate room temperature and a working lavatory.

When the aircraft enters the cruise phase after takeoff and climb, it is controlled by the Air Route Traffic Control Center (ARTCC) [32]. The ARTCC handles a vast amount of traffics, and most of IFR flights are under the control of the ARTCC. It is responsible for ensuring a safe route to all aircraft within the airspace, including redirecting aircraft from bad weather. The ARTCC is managed by dozens of people and has a limit on the number of flights that they can handle. This can lead to bottlenecks and delays. In most cases, however, the aircraft that are behind schedule at this stage can speed up and keep up with the schedule [33].

1.5.2 Categorization by Level of Details

According to the previous research, the main bottleneck that disrupts the ATC system is the major airports terminal airspace in the NAS [34]. In other words, the efficient operation of the terminal can play an important role in ATFM. Therefore, the scope of this thesis has narrowed down to the major airport's terminal area.

There are two approaches to analyzing the operations of airport terminal airspace: the macroscopic and the microscopic approach. In macroscopic scales, low levels of details are typically considered [35] and delays are aggregated at the airport level. From the macro point of view, the shortfall in airport capacity and a capacity-demand imbalance are the primary cause of delays. Besides, the microscopic approach considers each aircraft individually and reproduce as precisely as possible [35]. Micro reasons for air traffic delays are inclement weather, mechanical problem and the operational issues [36]. Even with all the known causal factors, macroscale delays cannot be reliably predicted from microscale information [33] and vice versa. To increase the reliability of the analysis, the microscale and macroscale must be separately analyzed.

Problem Statement

In order to have efficient and effective air transportation system, we need to analyze terminal area air traffic flow from macro and micro point of view.

Macro-level Causes

A definition of airport capacity given by Reference [37] is:

> *The hourly departure or arrival throughput that an airport can sustain during periods of high demand*

Also, a definition of airport demand can be given as follows:

> *Current or historical flight schedule*

Air Traffic Flow Management (ATFM) seeks to maximize airport throughput, balancing increasing aviation demand and system capacity. By doing this, it can also maximize the utilization of the NAS. However, maintaining a balance between demand and capacity is tricky. The number of departure and arrival flight is planned to be below the capacity level, but often the capacity is exceeded. For example, the maximum capacity may decrease if the terminal weather is bad or if an event occurs.

What happens when the number of scheduled flight exceeds the departure/arrival capacity? As air traffic demand approaches the capacity of the air traffic network, the regular or published schedules become difficult to operate normally. Moreover, if demand is close to the airport capacity, severe airline delays can occur [38]. Details are in Figure 1.10. Airport operations are stable when demand remains low relative to the capacity. As demand approaches the capacity of an airport, delays increase nonlinearly. Because flight schedules are optimized and tightly organized, it is not easy to recover to a normal state if there is a perturbation in the schedule [18]. The delayed flights form a long queue and the delay minutes will eventually diverge causing chaos at the airport. One thing to note from the figure is that you can not eliminate delays even if the airport's capacity improves in the future [25].

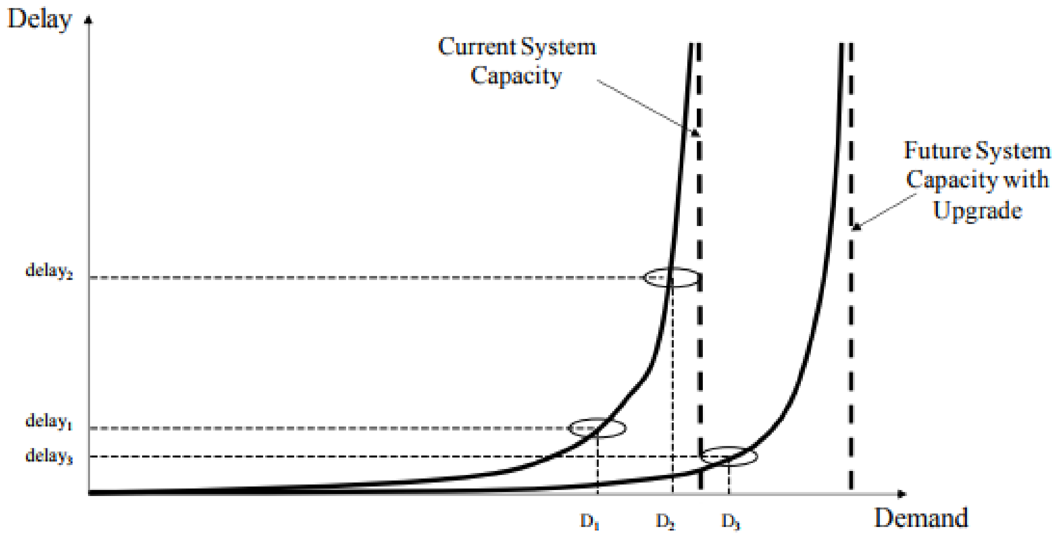


Figure 1.10: The relationship between delay, demand and capacity [25]

According to the data released by Eurocontrol in 2013, delays associated with increased demand and capacity interactions will rise from 1.2 min/flight in 2016 to 6.3 min/flight in 2040, transforming them from a minor to a major contributor of the total delay [39]. Some people may think that by building extra runways, air traffic congestion can be relieved. However, increasing air traffic capacity in this way may only be a temporary fix as demand rises continuously. It also requires enormous capital investment and time to implement. A complementary method of reducing airport congestion is to improve air traffic management technologies. Predicting capacity changes and managing air traffic flow accordingly can provide a quicker and cheaper alternative to mitigate airport congestion. If the change in capacity can be predicted in advance, the number of flights waiting in the queue by adjusting the departure and arrival pace.

The applicability of the capacity prediction is not limited to here. Many of advanced traffic management models assume that airport capacities are known. Therefore, it is important to accurately predict airport capacity to achieve superior performance of the airspace simulation model. Also, air traffic management (ATM) highly depends on the accurate predictions of air traffic flow [40]. If the air traffic flow is estimated accurately, air traffic management (ATM) can be more effective in the following tasks [41] [40] [42]:

- Identifying air traffic problems in the first place
- Addressing and mitigating the adverse effects of air traffic problems timely
- Avoiding the potential risks in the first place
- Assisting the controllers in taking more efficient decisions

Furthermore, traffic flow prediction in a terminal area is regarded as a critical element for [43]

- Advanced traveller information systems,
- Advanced traffic management systems,
- Advanced air traffic systems, and
- Commercial airline operations.

Based on these observations, the first observation can be drawn as follows:

Observation 1

From a macro perspective, there is a potential benefit to be gained by improving the predictive performance of air traffic flows at the airport level.

Micro-level Causes

Airlines seek efficiency and profitability in their operations [44]. ATC and passengers also want efficient operation of the air traffic system, but airport congestion hinders them. This necessity has led to air traffic scheduling to manage airport traffic congestion. However, air traffic scheduling is not enough. The reason is that a chain of complicated events occurs before an aircraft departs and any one of them can lead to unexpected delays [45], as in Chapter 1.5.1. Indeed, the actual operating schedule often deviates from the air traffic schedule due to unexpected accidents.

The Bureau of Transportation (BTS) has categorized airline delays into the following five main causes [46] and their percentage share are shown in Figure 1.11 :

- Air Carrier: The cause of delays was due to circumstances within the airline's control (e.g. maintenance or crew problems, aircraft cleaning, baggage loading, fuel).
- Extreme Weather: Significant meteorological conditions (actual or forecasted) that, in the judgment of the carrier, delays or prevents the operation of a flight such as a tornado, blizzard or hurricane
- National Aviation System: Delays and cancellations attributable to the national aviation system that refers to a broad set of conditions, such as non-extreme weather conditions, airport operations, heavy traffic volume, and air traffic control
- Late-arriving Aircraft: A previous flight with the same aircraft arrived late, causing

the present flight to depart late.

- Security: Delays caused by the evacuation of a terminal or concourse, re-boarding of aircraft because of a security breach, inoperative screening equipment or long lines in excess of 29 minutes at screening areas. Right after the September 11 attacks, the FAA halted all 36000 to 40000 flights taking off from the United States [47].

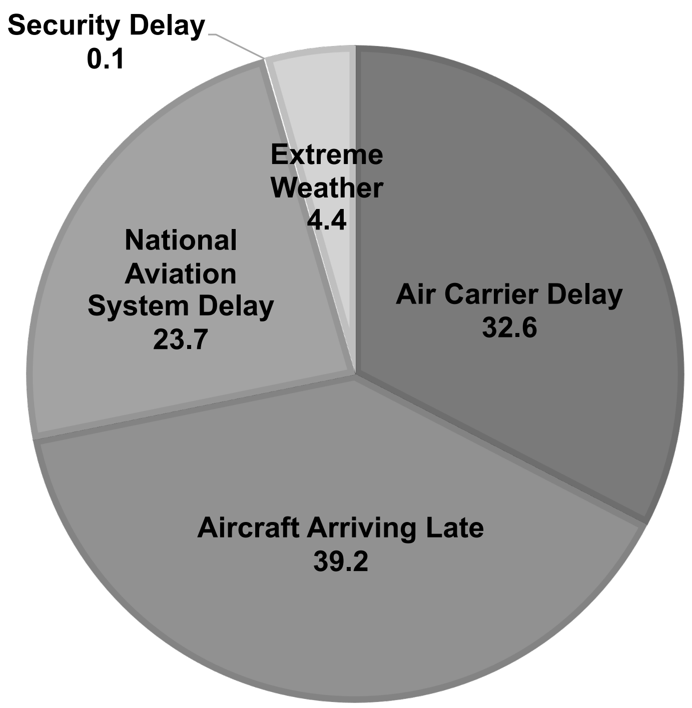


Figure 1.11: Causes of Flight Delays and the Percentage of Total Delay Minutes in 2016 [24]

The most notable of the five causes is the weather. The weather is not only one of the five factors of delays but is also closely connected with other factors. For example, the National Aviation System category can include delays due to the re-routing of flights by inclement weather. Besides, the weather is also a factor affecting late-arriving aircraft although airlines dont report the causes as the weather. By considering those facts, weather's percentage share accounts for about 40% of the total delay minutes [46]. Figure 1.12 shows the percentage of weather-related delays for a decade. There is even more extreme data. According to the Operations Network (OPSNET), the official source of the NAS operations and delays data, the largest cause of air traffic delay is the weather. The OPSNET standard ”Delay by Cause” report recorded that 69% of delays from 2008 to 2013 are caused by weather [48].

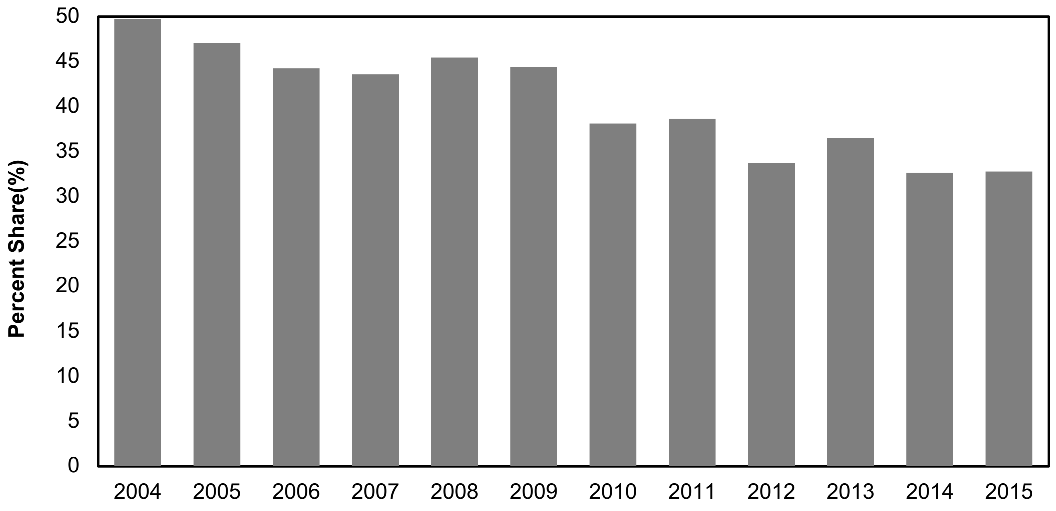


Figure 1.12: Weather-related Delays' Percent of Total Delay Minutes [46]

Disturbances to the air traffic schedule are inevitable because the occurrence of events that affect the schedule is out of control. Moreover, delays often occur because of a combination of causes. So there is always variability in the actual operating schedule. Therefore, it is necessary to analyze the patterns of occurrence and to study the countermeasures to mitigate airline delays.

The objective of air traffic schedule prediction is to [43]:

- Help users make better travel decisions,

- Alleviate traffic congestion,
- Reduce carbon emissions, and
- Improve traffic operation efficiency.

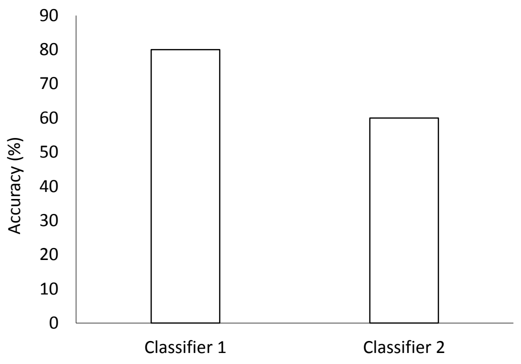
With accurate delays predictions, passengers can prepare for disruptions of their journey and airlines can actively respond to the potential causes of flight delays and mitigate their impact. By doing this, airlines will be able to maintain customer loyalty, enhance customer service and secure competitiveness. Also, accurate predictions reduce possible losses with much less time and effort. These reasons enable the formulation of another key observation:

Observation 2

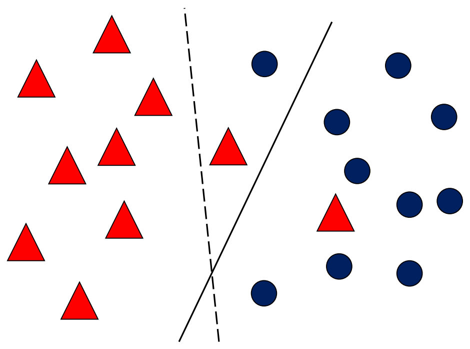
From a micro point of view, an accurate analysis of the delays taking into account the relationship with weather is needed.

1.6 Cost-sensitive Learning

Suppose you are going to build a classification model using the general algorithm to find the occurrence of delays. Most classification learning algorithms are designed to minimize the expected number of misclassification errors. Therefore, the most common criterion for evaluating the performance of a classifier is accuracy. In Figure 1.13(a), one can say that Classifier 1 is a better classifier based on accuracy. However, the case of Figure 1.13(b) is not that simple. In Figure 1.13(b), the solid line and the dotted line are classifiers for classifying the red triangles and the blue circles. Those two classifiers correctly classified 18 samples out of 20. Then can we conclude that the dotted and the solid classifiers have the same performance? What if the red triangles are more critical class than the blue circles and they cause a bigger cost when it is misclassified?



(a)



(b)

Figure 1.13: Example of Classifiers

Classification problems are inherently associated with misclassification costs. Flight delays prediction also is no exception. It involves misclassification costs. One thing to note is that in many real-world problems, the cost of misclassification is asymmetric. Not recognizing samples of one class is more expensive than the other. However, classifiers are usually trained under the assumption that all types of misclassification costs are the same

[49]. Unless otherwise noted, asymmetric misclassification cost is not taken into account. Classifiers built upon the assumption of equal misclassification costs will not perform well [50]. Models can provide meaningful predictions on the problem only when they are built with the cost of classification in mind. This motivates the adaptation of cost-sensitive learning.

1.6.1 Asymmetric Costs of Delays Prediction

In a binary classification problem, there are two types of error distinguished: a false positive error and a false negative error. In the problem of predicting flight delays, classifying an on-time flight as delayed is a false positive error. On the other hand, the number of delayed flights that are not captured by the model corresponds to a false negative error.

Table 1.2: Confusion Matrix of a Binary Classification Problem

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

In a binary classification problem, all cases that can occur depending on the actual class and the prediction class are shown in Table 1.2. The cost of a binary classification is a function of the actual and the predicted class.

$$cost_{predicted,actual} = f(class_{predicted}, class_{actual})$$

Thus, the cost of each case of Table 1.2 can also be expressed as Table 1.3. The cost of correct classification among the four costs of Table 1.3 should be less than the cost of the

Table 1.3: Cost Matrix of a Binary Classification Problem

	Predicted On-time	Predicted Delayed
Actual On-time	$cost_{on-time,on-time}$	$cost_{delayed,on-time}$
Actual Delayed	$cost_{on-time,delayed}$	$cost_{delayed,delayed}$

incorrect classification. In other words, the following two conditions must be met:

$$cost_{delayed,on-time} > cost_{on-time,on-time} \tag{1.1a}$$

$$cost_{on-time,delayed} > cost_{delayed,delayed}. \tag{1.1b}$$

These conditions are for the fairness of the classifiers trained with cost-sensitive learning, and there may be the following problems when they are violated [51].

- When Equation (1.1a) is violated:
  $cost_{delayed,on-time} \leq cost_{on-time,on-time}$ but $cost_{on-time,delayed} > cost_{delayed,delayed}$. This means that the cost of misclassifying on-time samples is less than that of properly classifying them. In terms of cost-sensitive learning, it is advantageous to classify on-time samples as delayed. Therefore, a cost-sensitive classifier will attempt to classify all samples as delayed regardless of the actual class of the samples. This result negates the purpose of the classifier itself and therefore shows that Equation (1.1a) must be satisfied.

- When Equation (1.1b) is violated:
  $cost_{on-time,delayed} \leq cost_{delayed,delayed}$ but $cost_{delayed,on-time} > cost_{on-time,on-time}$.

Violation of Equation (1.1b) implies that the cost of classifying the delayed sample as on-time is cheaper than the cost of correctly classifying it. Therefore, the classifier trained by cost-sensitive learning will attempt to classify all samples into the on-time class. Equation (1.1b) must also be met for the same reason as Equation (1.1a).

Because Equation (1.1a) and (1.1b) have a clear basis as described above, this study assumes that the correct classification is always cheaper than the incorrect classification. For these two conditions, Margineantu presented another interpretation. It is a condition that no column can dominate another column. In Table 1.3, if the value of column i is less than the value of column j for all the rows, the j^{th} column will never be predicted [52]. Let us look at an example case of Table 1.4. According to Table 1.4, it is cheaper to predict the sample as the negative class, whatever the actual class is. Therefore, ignoring the positive class will be the best policy. If this is the case, no learning is needed. So it will be left out of the discussion.

Table 1.4: Example Cost Matrix where One Class Label is Always Ignored

	Predicted Negative	Predicted Positive
Actual Negative	1	2
Actual Positive	0.5	1

In most studies, the goal is to evaluate the performance of cost-sensitive classifiers for situations where the actual cost matrix is not provided [52]. Therefore, a series of cost matrices must be generated based on some rules. In this thesis, it is assumed that accurate predictions are not costly at all:

$$cost_{on-time,on-time} = 0 \tag{1.2}$$

$$cost_{delayed,delayed} = 0. \tag{1.3}$$

This assumption can ensure compliance with Equation (1.1a), Equation (1.1b) and the non-dominance of cost matrix columns.

The cost matrix generally has the following characteristics:

- Multiplying each element of a given cost matrix by a positive constant does not change the optimal policy of the cost matrix [51].

- Adding a constant to each element of a given cost matrix does not change the optimal policy of the cost matrix [51].

With these properties of the cost matrix, any given cost matrix can be transformed into the form of Table 1.5. In addition, the cost derived from the correct predictions is beyond the scope of this thesis, so the value c'_{11} in Table 1.5 is also ignored. Then, for the remaining two items, the cost of the false negative error can be expressed as a ratio of the cost of the false positive error.

Table 1.5: Simple Cost Matrix Form

	Predicted Negative	Predicted Positive
Actual Negative	0	1
Actual Positive	c'_{01}	c'_{11}

The cost here is not a loss from the actual class, but an opportunity missed due to incorrect predictions [51]. It uses the state before the prediction as a baseline and measures how much the wrong prediction caused a loss compared to the baseline. In that respect, a false negative error is more expensive compared to a false positive error in many cases. Similarly, not recognizing delayed flights causes more significant losses than on-time flights in this problem.

If flight delays can be predicted correctly, the Air Traffic Control (ATC) can establish an alternative plan to coordinate air traffic and minimize possible congestion. On the other hand, if delayed flights are misclassified, this opportunity will be lost and more congestion will occur at the airport. Moreover, flight delays without prior notice may reduce the airline's customer loyalty. In addition, passengers can also suffer economic losses due to unforeseen delays. On the contrary, the monetary loss of incorrectly predicting on-time flights is quite less compared to the opposite case. The misclassification costs of the false positive error are from degraded airport efficiency by preparing unnecessary countermeasures and unnecessary passengers concerns. Considering all of these facts, we can express the following conclusions about the cost of the false positive error and the false negative error:

$$cost_{on-time,delayed} \gg cost_{delayed,on-time}$$

Classifiers trained without any information about misclassification costs will be of no practical use. This is because that they are based on the false assumption that the misclassification cost is constant. We should pay particular attention to the cost of misclassification when the cost imbalance is substantial. This leads to the third observation:

Observation 3

Classifiers for individual flight delays that assume the same misclassification costs are not practical.

1.7 Research Objective

Aiming at an efficient and effective air transportation system, the observations of the present system lead to research objective as follows:

Research Objective

To develop a methodology that can efficiently and practically predict macro and micro-level air traffic flow in the terminal area

The whole research process involved in establishing this research objective is outlined in Figure 1.14. The first observation motivates improvement of hourly airport capacity prediction. The second observation encourages to focus on the on-time performance of individual flights. The last observation emphasizes the importance of taking into account misclassification cost when building a predictive model for the individual delays.

For each research area, research question, problem settings, methodologies, and experiments results will be explained. In Chapter 2, a detailed approach to airport capacity prediction is presented with a review of previous studies. The performance of the proposed method is analyzed, and its performance is compared with that of other methods. Chapter 3 concentrates on flight delays prediction problem. Flight delays data is examined closely and methods to improve predictability are discussed. Chapter 4 provides background and methods about cost-sensitive learning. This chapter describes how to implement cost-sensitive learning and its performance. Chapter 5 closes with a concluding remark.

This thesis collates material from 2 conference papers [53] [54] and a journal article under review by the author. The contents of Chapter 2 come from the journal article. Chapter 3 is based on Reference [53] and Chapter 4 uses material from Reference [54]. Some material from each paper is also used in this introductory chapter.

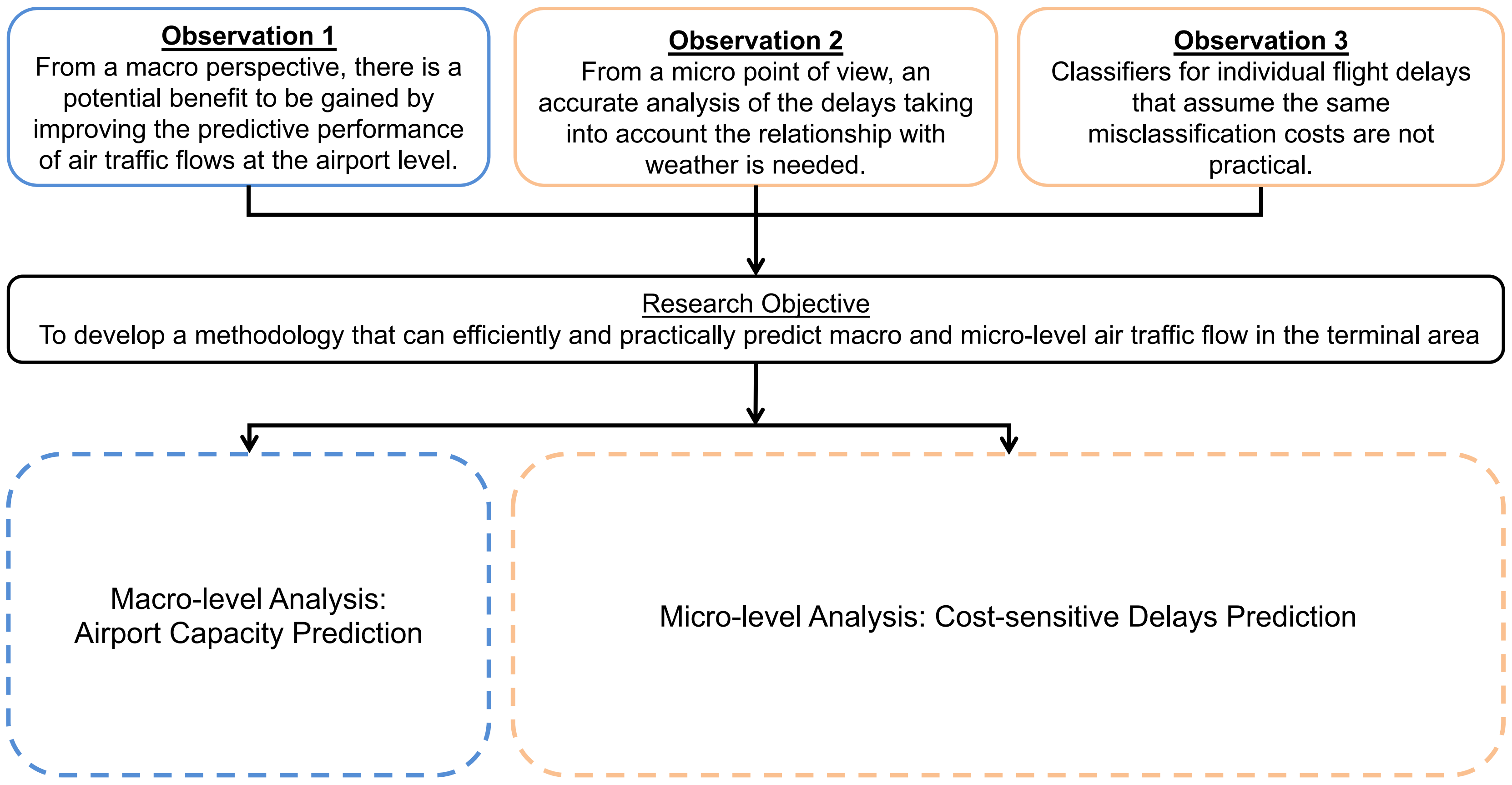

Observation 1
From a macro perspective, there is a potential benefit to be gained by improving the predictive performance of air traffic flows at the airport level.
Observation 2
From a micro point of view, an accurate analysis of the delays taking into account the relationship with weather is needed.
Observation 3
Classifiers for individual flight delays that assume the same misclassification costs are not practical.
Research Objective
To develop a methodology that can efficiently and practically predict macro and micro-level air traffic flow in the terminal area
Macro-level Analysis: Airport Capacity Prediction
Micro-level Analysis: Cost-sensitive Delays Prediction

Figure 1.14: Establishment of the Research Objective

CHAPTER 2
AIRPORT CAPACITY PREDICTION

2.1 Backgrounds

Accurate prediction of airports capacity is important for air traffic flow management. Both overpredictions and underpredictions will cause en-route and terminal delays [55]. From analytical methods to simulations, there have been several approaches to predict airport capacity accurately as seen in Figure 2.1. The methods are listed according to the degree of precision and the cost required to use the model. From left to right, costs increase, and precision improves. In the following sections, we will identify the features of these methods and discuss possible improvements.

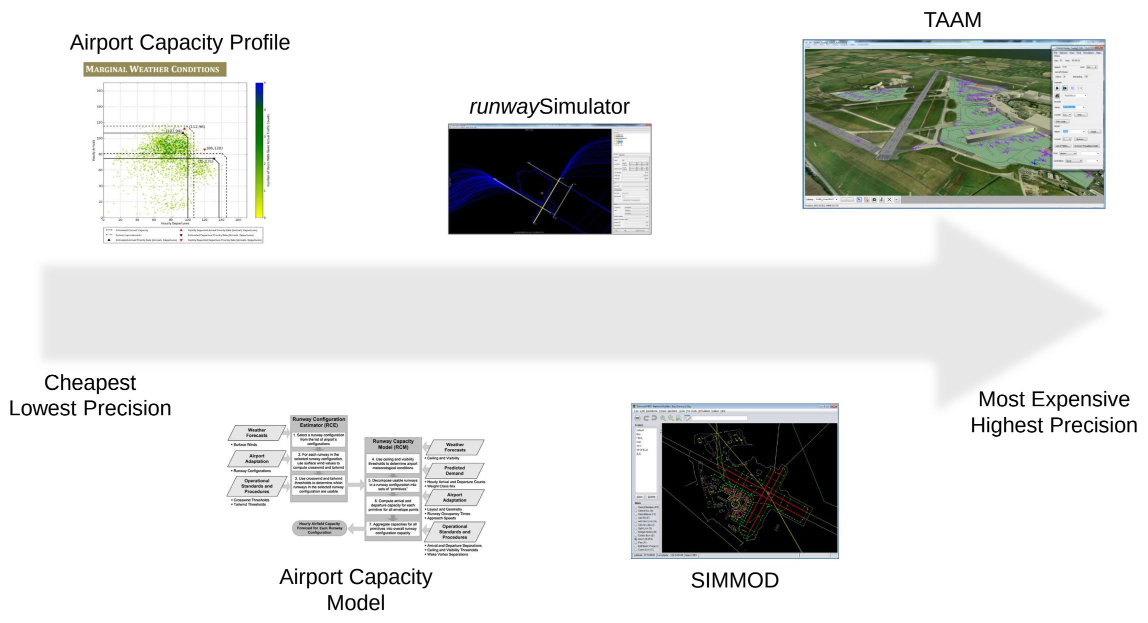
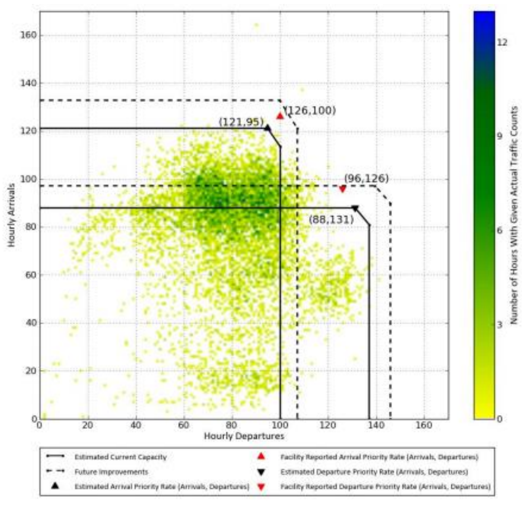


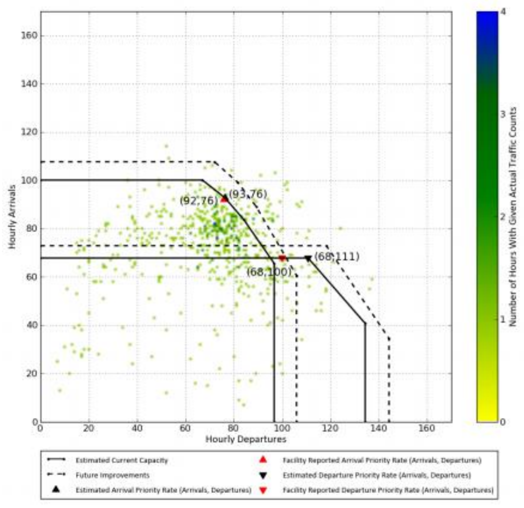
Figure 2.1: Approaches to Predict Airport Capacity

2.1.1 Airport Capacity Profile

The airport capacity profile developed by the Federal Aviation Administration shows the hourly throughput that an airport is able to sustain during periods of high demand [37]. The airport capacity profile is provided for each airport and can be used to identify airport capacity at a glance. Examples of the airport capacity profile are presented in Figure 2.2. The frontier of the figure is estimated from $runway$Simulator described in Section 2.1.3. Each point corresponds to the number of actual arrivals and departures per hour between 7 am and 11 pm local time from October 2008 to September 2010. Figure 2.2(a) shows the case when the visual flight rule is secured. Figure 2.2(b) is a case where IFR controls flights due to inclement weather conditions. Because of the reduced capacity due to inclement weather, we can see that the points in Figure 2.2(b) are at the lower left in comparison with Figure 2.2(a).



(a) Visual Weather Conditions



(b) Low Instrument Weather Conditions

Figure 2.2: Sample Airport Capacity Profile of ATL [37]

2.1.2 Analytical Model

The second approach to predict airport capacity is an analytical model. This analytical model uses mathematical and statistical representations of airport operations [56]. The federal aviation administration (FAA) has created an analytical model for the airport capacity prediction called Airfield Capacity Model (ACM). The early version of this analytical model can determine the capacity of the individual elements that make up the airport [57]. The capacity of each element is determined by the closed form equation, which is the inverse of the average service time of all aircraft passing through the element.

Integrated Airport Capacity Model (IACM), an advanced version of ACM, provides a probabilistic forecast of airport departure rates (ADR) and airport arrival rates (AAR) [55]. The schematic in Figure 2.3 explains the concept of IACM. The capacity estimates of IACM are based on the runway capacity module and the terminal airspace capacity module. The runway capacity module uses predicted demand, surface winds, ceiling height and visibility to predict runway capacity. On the other hand, the terminal airspace capacity module estimates the airspace capacity using convective weather forecast. IACM integrates the two modules in consideration of airport constraints like runway layout, operational standards and procedures. In contrast to the existing analytical models that only use deterministic weather, IACM utilizes real-time weather forecasts as input, resulting in a probability distribution due to uncertainty [58] [59].

Both the strengths and weaknesses of the analytical model come from its macroscopic characteristics. In general, analytical models use mathematical and statistical representations of airport operations [56]. It is fast and computationally non-intensive because the analytical models make use of relatively abstract and simplified expression. For the same reason, the analytical models are with low-level of details. The more complex the problem, the more difficult it is to apply the analytical models, or the more imperfect the representation of the system.

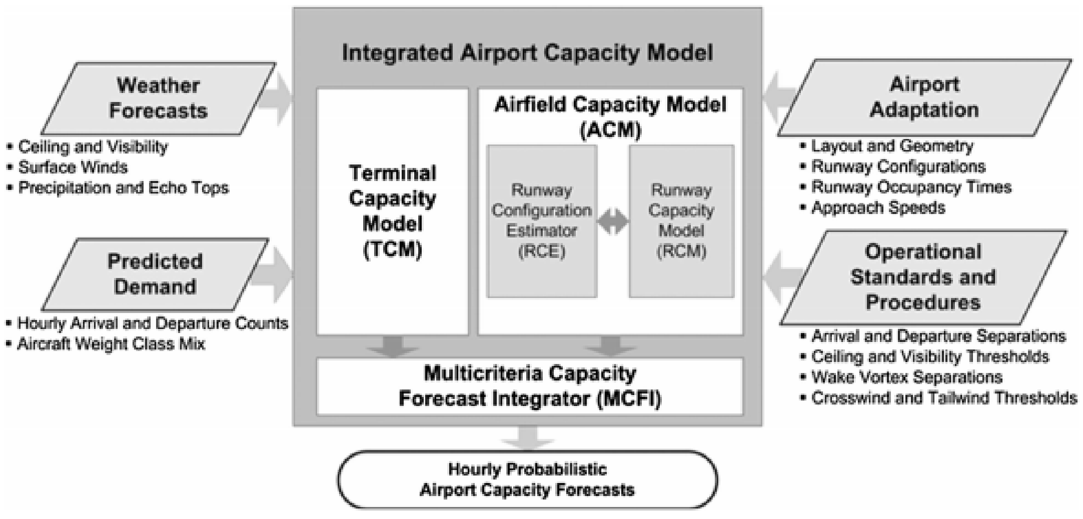

Integrated Airport Capacity Model
Weather Forecasts
• Ceiling and Visibility
• Surface Winds
• Precipitation and Echo Tops
Predicted Demand
• Hourly Arrival and Departure Counts
• Aircraft Weight Class Mix
Terminal Capacity Model (TCM)
Airfield Capacity Model (ACM)
Runway Configuration Estimator (RCE)
Runway Capacity Model (RCM)
Multicriteria Capacity Forecast Integrator (MCFI)
Hourly Probabilistic Airport Capacity Forecasts
Airport Adaptation
• Layout and Geometry
• Runway Configurations
• Runway Occupancy Times
• Approach Speeds
Operational Standards and Procedures
• Arrival and Departure Separations
• Ceiling and Visibility Thresholds
• Wake Vortex Separations
• Crosswind and Tailwind Thresholds

Figure 2.3: Integrated Airfield Capacity Model (IACM) [55]

2.1.3 Simulation-based Model

There also has been a simulation-based approach. One of the simulation models, Total Airspace and Airport Modeller (TAAM), is a detailed simulation tool for modeling the entire air traffic system. TAMM requires an input file that describes airport layout, air traffic schedules, air traffic control rules, and so on. Using this, its internal algorithms and custom rules investigate airport and airspace usage and calculate relevant metrics [60].

Another example of a detailed simulation tool is SIMMOD [61]. The primary inputs required by SIMMOD are a specification of the network structure for the aerodrome and airspace being simulated. It also needs a description of the traffic to travel within the network, such as the flight path and the path between the gate and the runway. SIMMOD simulates airport layout components as nodes, and procedure connecting two components as links [62].

While TAMM and SIMMOD are high-resolution simulation tools, *runway*Simulator is a medium-resolution simulation capability [63]. It is developed by the MITRE Corporation to bridge the gap between high-level analytical models and detailed simulation models [64]. *runway*Simulator calculates airport capacity by taking airport configuration data and flight data as in the previous models. However, it differs in that it requires less effort and assumptions for simulation.

These simulation-based models create virtual components of airport operations and simulate their flow [65]. They then estimate the flow at each component [55]. Thus, they can analyze operations through the airspace, taxiways and gate/ramp areas [66]. The capacity determined by the simulation-based model is the maximum sustainable rate at which the airport can meet the demand for a specific aircraft mix, following the separation requirements in a given runway configuration [67]. Thus, an analyst can select the scenario they want to analyze and do a sophisticated analysis accordingly [66]. Also, wholly new or unusual runway layout and operation can be modeled and evaluated. However, simulation-based models require the labor-intensive setup. A user should specify details such as runway layout, type of aircraft, separation requirements [68].

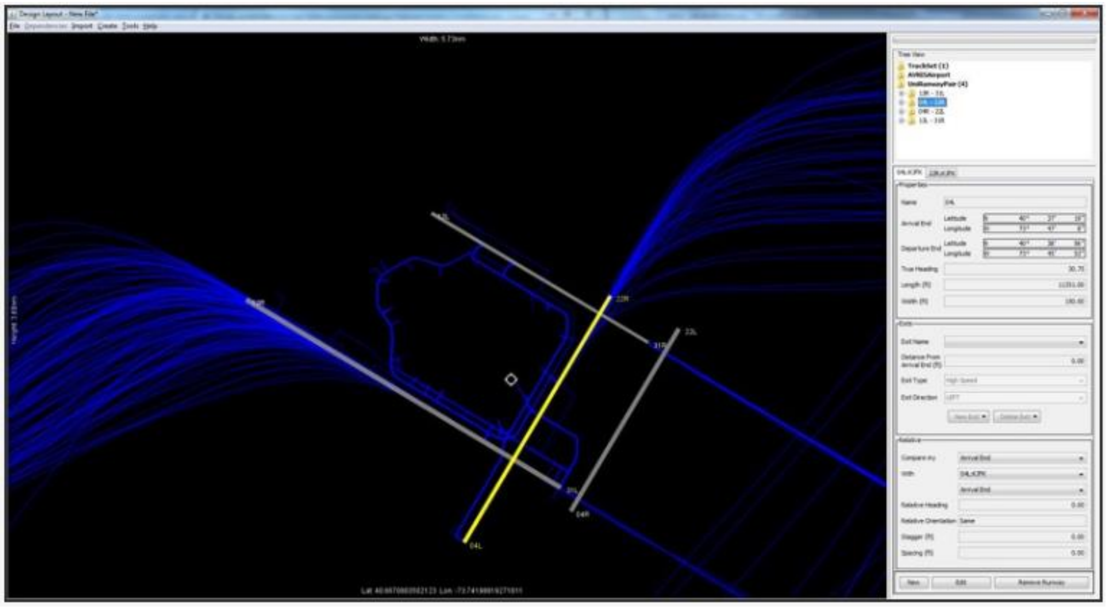


Figure 2.4: Screenshot of *runway*Simulator [69]

2.2 Problem Definition

The airport capacity prediction is important in air traffic flow management [40]. This is because many of Advanced traffic management models assume that airport capacities are known. Achieving accurate airport capacity prediction, however, is difficult as it depends on many interrelated factors, including operational standards and procedures, runway configuration and status, meteorological conditions, and expected air traffic demand mix [59]. Therefore, to accurately predict airport capacity, it is advisable to choose the right method, taking into account the characteristics of the problem. The characteristics of the problem considered in this study are described below.

First, the airport capacity throughput data has a daily pattern. Figure 2.5 represents the weekly arrival capacity obtained from the Aviation System Performance Metrics (ASPM). It has a periodic pattern. Usually, the operational capacity of airports is contingent on demands. Since passengers do not like to fly too early or late at night, the schedule provided by airlines often makes a peak during the afternoon to evening hours. It is very low in the early morning and late night. Forecasts for these patterns are based on time series analysis. The information needed to infer capacity at the next time step can be obtained from the current capacity level.

The second characteristic is nonlinearity. In general, airport capacity is expressed in terms of two interdependent terms, departure capacity and arrival capacity [59]. Arrival capacity and departure capacity are nonlinearly related as shown in Figure 2.6. Their relation is nonlinear as airport facilities such as runways must be appropriately allocated on departure and arrival. The nonlinear relationship between them is characterized by three or more cases that determine how to allocate runways. Each case corresponds to the vertex of the frontier in Figure 2.6. It can be either fully departures, or either arrivals, or balanced departures and arrivals. To maximize capacity, some of the busy airports such as ATL, EWR or JFK tend to operate in an arrival or departure priority mode, as opposed to a balanced

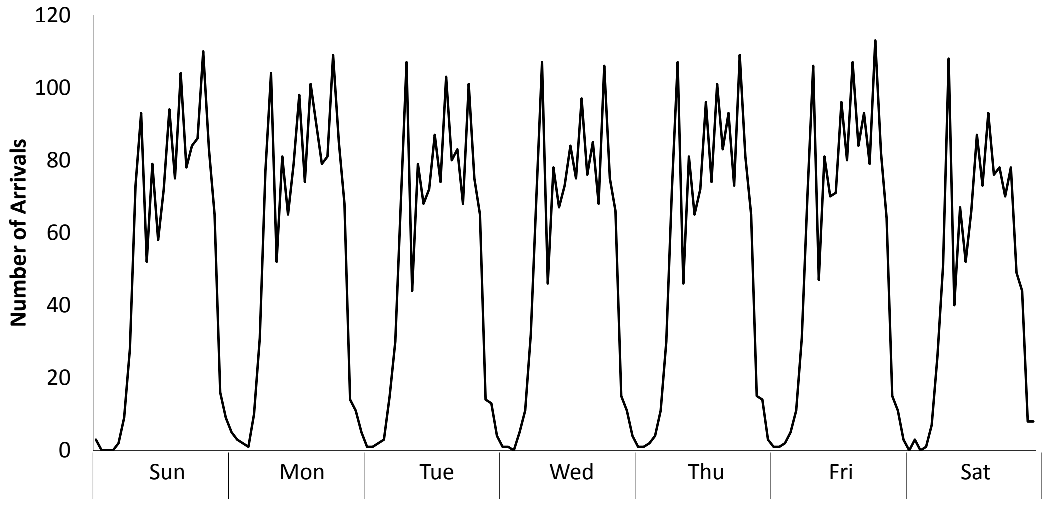


Figure 2.5: Airport Arrival Rate of ATL during 1/4/2009 - 1/10/2009 [70]

operation. An arrival or departure priority operation is only feasible when the airports flight schedule is unbalanced for sustained periods of time.

The last thing to consider is that the airport capacity changes in response to weather. As insufficient visibility tightens restrictions on takeoff and landing, takeoff capacity and landing capacity of an airport are reduced. Aircraft arriving in excess of capacity will not be able to land and will have to be diverted to an alternative airport. Also, departing aircraft must wait in a queue stuck on the ground until the situation improve. In many of the major airports, the application of IFR separation and avoidance procedures are the major constraints [71].

According to the International Civil Aviation Organization (ICAO)'s weather classification, visual meteorological conditions (VMC) refers to fair to good weather. Additionally, instrument meteorological conditions (IMC) is generally when the visibility falls below three statute miles (SM) or the ceiling is less than 1000 feet above ground level (AGL) [72]. The weather conditions such as visibility and ceiling lead to airport capacity. The weather

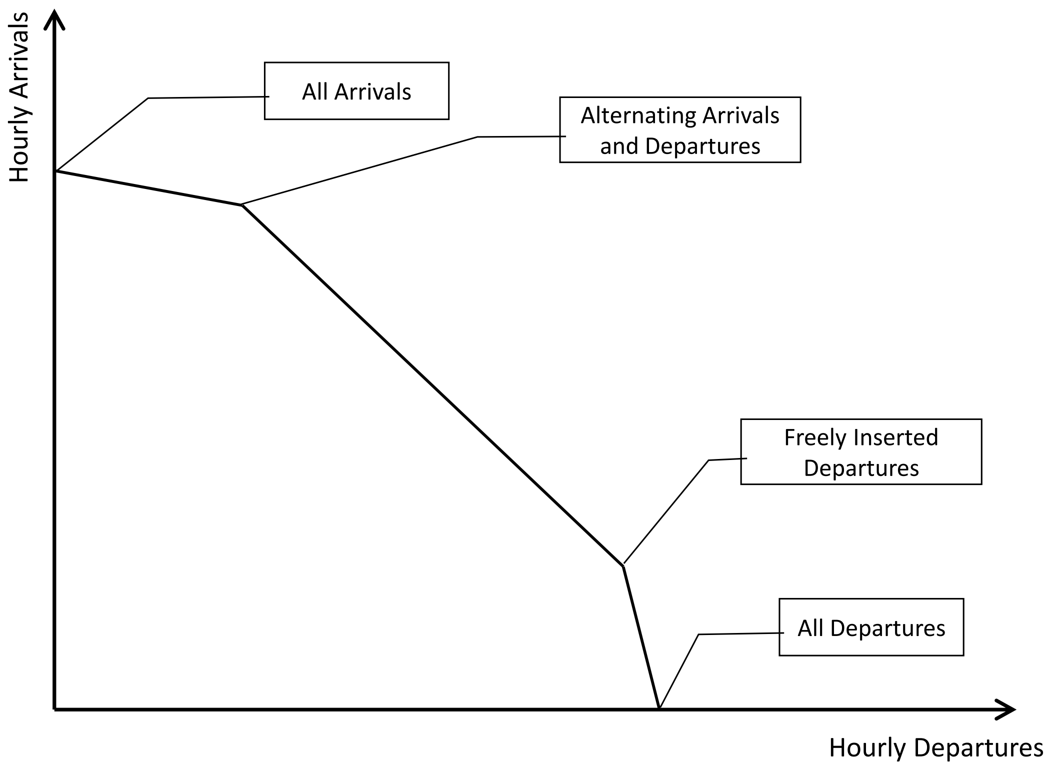


Figure 2.6: Nominal Airport Capacity Profile [59]

conditions determine whether the flight rules are Visual flight rules (VFR) or Instrument Flight Rule (IFR). Depending on which rule the flight is controlled, the separation distance between the aircraft is also affected by [72]. Detailed criteria for VMC and corresponding flight rules are given in Appendix B.1 and Appendix B.2.

For the two consecutive arriving aircraft, the operating rules depend on the visibility conditions. Visual Flight Rules (VFR) generally regulates flights operated at VMC. An aircraft will operate under Instrument Flight Rule (IFR) during IMC [73]. Under VFR, the pilot is responsible for separation from other traffic by seeing and avoiding [74] [72]. The separation requirements applied in VFR are mainly about the minimum distance to avoid collisions and wake turbulence [71]. On the other hand, the air traffic control (ATC) takes control of the airspace and is responsible for the separation between aircraft during IFR

[72]. Also, additional separation requirements apply to the common access path under IFR [73]. For example, the minimum separation requirement between parallel centerlines for simultaneous landings and takeoffs using VFR operation is $700\ ft$ while $4300\ ft$ when using IFR [75]. Runways are closely spaced because of the lack of ground resources, and some of the parallel runways may not be available because they do not meet the tightened separation requirements under the IMC. This causes flights that fail to meet scheduled schedules due to lack of resources.

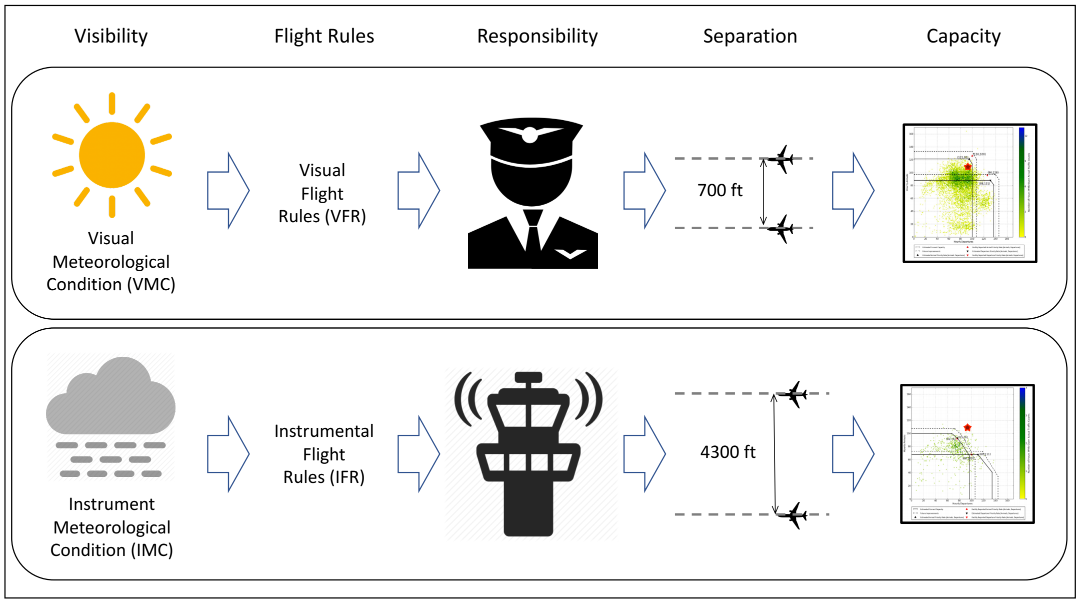

Visibility
Flight Rules
Responsibility
Separation
Capacity
Visual Meteorological Condition (VMC)
Visual Flight Rules (VFR)
700 ft
Instrument Meteorological Condition (IMC)
Instrumental Flight Rules (IFR)
4300 ft

Figure 2.7: Example Case of Capacity Reduction by Low Visibility

In the above paragraphs, we have seen how visibility is connected to delays by reducing airport capacity. The magnitude of the capacity change due to weather conditions varies significantly from the airport to airport. The weather heavily influences some airports, and there is a big difference in their capacity depending on whether the flight rules are VFR or IFR. In Figure 2.8, the meteorological characteristics and the degree of the capacity change for the major airports in the NAS are shown. The red bars present capacity degradation from VMC to IMC and the blue bars indicate the percentage of time that visibility/ceiling is poor at airport terminal area [66]. IMC reduce the capacity of an airport by about 30% on average from VMC. During IMC, capacity reduction occurs because extra separation requirements between aircraft cannot be met, or certain runways become unavailable. Airports with huge differences between Visual and Instrument condition regularly experience significant delays in bad weather.

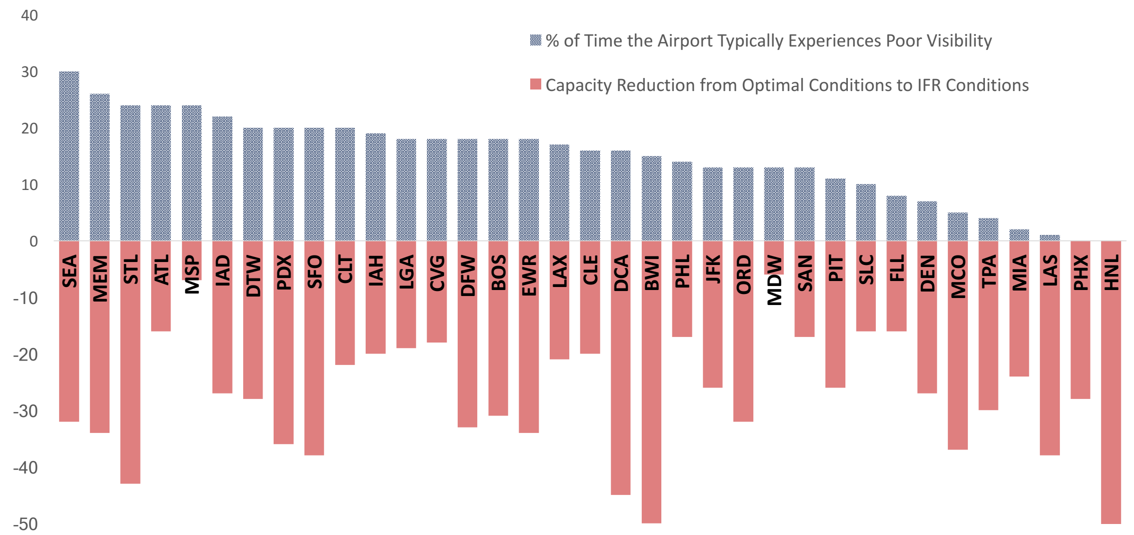


Figure 2.8: Airport Capacity Loss vs. Occurrence of Inclement Weather Conditions [66]

By focusing on those facts, the first research question is introduced.

Research Question 1

What is an accurate and computationally efficient method to predict the hourly capacity of an airport?

2.2.1 Possible Improvements with Artificial Neural Networks

The gap to be filled in existing methods is identified as follows:

- Labor-intensive Setup
- Mathematical Assumptions
- The Necessity of Detailed Information about Fleets, Runway Operation, etc.
- Significant Computational Cost

In recent years, the availability of air traffic data has improved significantly, and it has become a new turning point in air traffic analysis. A good example is the FAA aviation system performance metrics (ASPM) system. The airport analysis module of the ASPM provides information on aircraft departure and arrival rates for selected airports. It also includes information about hourly flight delays for each airport [70]. Since access to the advanced air traffic database has become more comfortable, studies have been proposed over recent years to improve the fidelity of models by presenting future air traffic models in a data-driven direction [76], [53], [54].

Among data-driven models, artificial neural networks model is particularly suitable to cope with the historical data accumulated over decades [77]. Its ability to extract information from data and discover correlations between functions is excellent. Artificial neural networks model also can provide a various level of details within reasonable training time if it is trained with proper data. Simulation methods that can provide the most detailed description are not always the best. It is because such high fidelity that provided in the simulation is not needed for certain purposes. Another important fact is that this method outperforms other methods in many applications when the right method is chosen and appropriately tuned. For these reasons, an artificial neural networks approach is being rapidly developed in recent years along with the improved data accessibility. Especially in previous studies, ground traffic flow prediction using artificial neural network was successful [43] [41] [78], so it is worth to test artificial neural networks algorithms to air traffic prediction.

Table 2.1: Comparison of Capacity Prediction Methods

Criterion	Polynomial Regression	Artificial Neural Networks	Simulation-based Models
Nonlinear data handling	×	○	○
Time-series data handling	×	○	○
Computational time	○	△	×

The previous studies of Section 2.1 and artificial neural networks are compared in Table 2.1 with the characteristics required by this problem. Given these characteristics of the capacity prediction problem and artificial neural networks, artificial neural networks seem to be a suitable method for the problem. This can be formulated as the first hypothesis:

Hypothesis 1

If suitable artificial neural networks are constructed using historical weather and airport traffic capacity data, then one could accurately predict the hourly capacity for a given airport under different scenarios.

2.3 Model Evaluation Criteria

To validate Hypothesis 1, experiment 1 is planned to be executed. For a comparative assessment of artificial neural networks algorithms, the accuracy and computational efficiency will be measured.

Here are some questions to be answered by experiment 1 in particular:

- Validation of optimal hyperparameters for each artificial neural networks architecture using Grid Search
- Demonstrate which method has the best accuracy
- Generalization of models trained with the data from specific airports

The performance metrics used to compare the performance of the models built in this paper are the mean squared error (MSE), the root mean square error (RMSE) and the mean relative error (MRE) commonly used in the regression analysis. The three performance metrics are defined as follows:

- Mean Absolute Error (MAE) $= \dfrac{1}{n}\sum_{i=1}^{n} |f_i - \hat{f}_i|$
- Mean Relative Error (MRE) $= \dfrac{1}{n}\sum_{i=1}^{n} \dfrac{|f_i - \hat{f}_i|}{f_i}$
- Root Mean Absolute Error (RMAE) $= \big[\dfrac{1}{n}\sum_{i=1}^{n}(|f_i - \hat{f}_i|)^2\big]^{\dfrac{1}{2}}$

where n is the number of samples, f_i is the ground truth value, and $\hat{f}_i$ is the value predicted by the model. Hypothesis 1 will be accepted if the artificial neural networks have improvements based on validation criteria.

2.4 History of Artificial Neural Networks

Artificial neural networks are a machine learning method using the principle of the human neural network. Artificial neural networks are widely used in leading companies such as Google and Amazon because it has excellent performance in various applications like image, voice recognition and natural language processing. Although artificial neural networks have recently emerged as a hot topic, the early concepts of artificial neural networks have already been suggested in the 1950s. In this section, we will look in chronological order to see what stages the artificial neural networks have gone through to reach the present.

The first idea of artificial neural networks was first proposed in 1943 by McCulloch and Pitts in their paper "A logical calculus of the ideas immanent in nervous activity" [79]. In this paper, the authors argued that human neural structures could be represented by networks consisting of zero and one transmissions. A more realistic algorithm was later presented in a paper by Rosenblatt [80]. Figure 2.9 is a perceptron that responds to visual stimuli visualized by Rosenblatt. The perceptron proposed in this paper is a feed forward classifier that computes the linear combination of n inputs and applies the activation function. The result of the activation function is a probability value, and depending on its value, the perceptron outputs a binary result of 1 or -1.

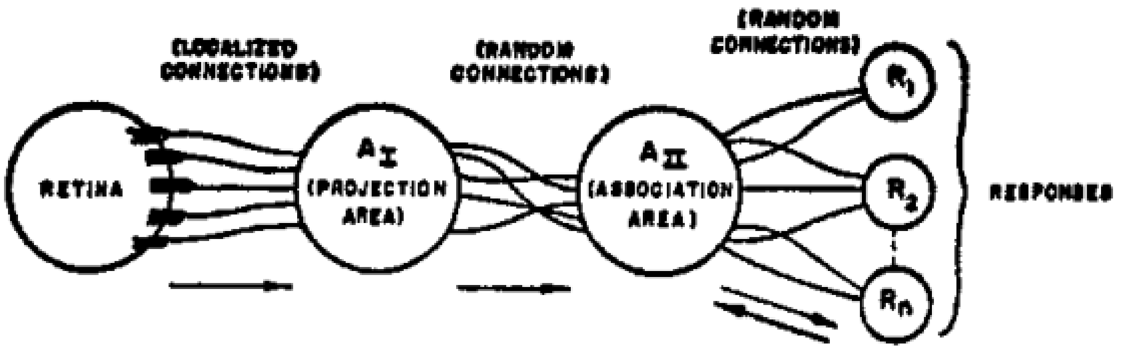


Figure 2.9: Organization of A Perceptron that Responds to Visual Stimuli [80]

After Marvin Minsky and Seymour Papert proved that the perceptron has limitations not being able to learn a simple XOR problem [81], the perceptron seemed to drift without further development. However, as the limitations of the perceptron are solved in a simple way, the development of artificial neural networks meets a big turning point. The proposed idea is multilayer perceptron that adds one or more hidden layers between the input and output layers. In addition to this idea, error back propagation was developed to enable multilayer perceptron learning [82]. This had made great progress in artificial neural networks by the early 90s as the artificial neural networks model became able to solve not only XOR but also more complex problems. In particular, Yann LeCun proposed Convolutional Neural Networks (CNNs) in 1998 and made a significant advance in image processing using artificial neural networks [83]. The CNNs are methods that make a hidden layer to process only local information, not global information and then combine the information in hidden layers downstream. LeCun showed the practicality of artificial neural networks by recognizing hand-written digits with LeNet, a topology using the CNNs.

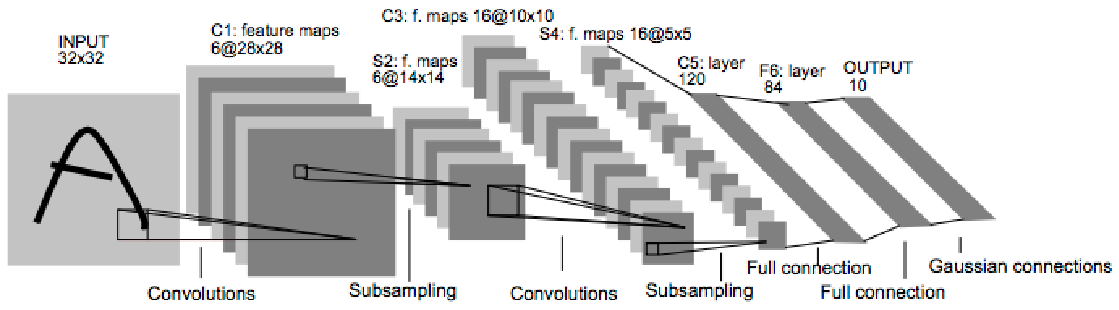


Figure 2.10: Architecture of LeNet-5 [83]

However, the research of artificial neural networks once again faced with limitations because of the vanishing gradient problem and the lower performance compared to simple machine learning algorithms such as Support Vector Machine (SVM) and Random Forest.

In the early 2000s, Geoffrey Hinton's work opens up a new era of artificial neural networks. Hinton showed that the artificial neural networks algorithms could produce good results without falling into local minima if the weights of each layer are pre-trained through unsupervised learning [84]. In addition, Bengio et al. have enabled the training of artificial neural networks algorithms by using autoencoder and made it possible to solve more complicated problems [85].

The development of artificial neural networks has gained more momentum as computing power has sharply increased and the amount of available data has increased enormously compared to the past. As a general purpose graphics processing unit (GPGPU) is used for the artificial neural networks algorithms, training time has been dramatically reduced. Thanks to data accessibility and GPGPU, artificial neural networks algorithms has begun to yield meaningful results from many applications.

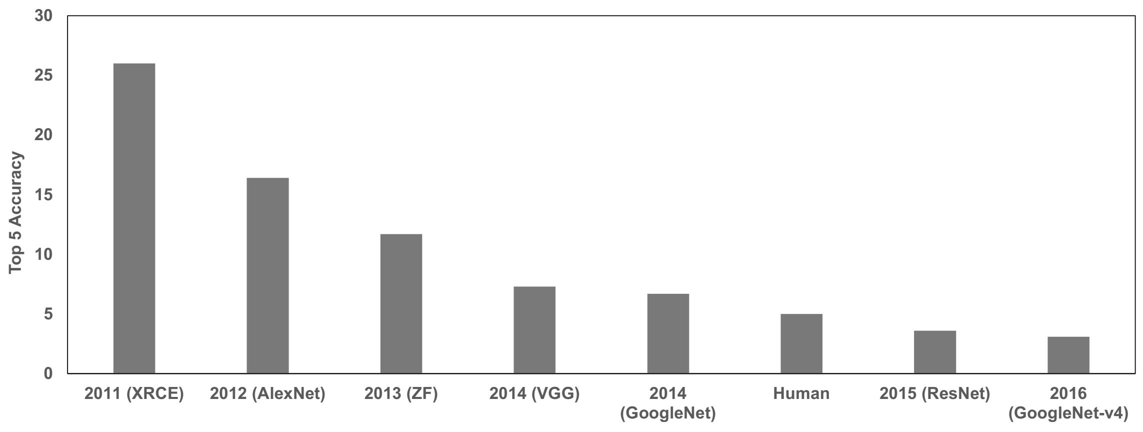


Figure 2.11: Winners and Their Top-5 accuracy of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2010-2016 [87]

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) has shown the great potential of artificial neural networks. ImageNet Large Scale Visual Recognition Challenge is a standardized benchmark for image classification and detection for millions of images [86]. It has been held annually since 2010, and more than 50 organizations have participated in this event. In image classification, teams perform multilabel classification to find the class which the sample is most likely to belong. In object localization, teams have the task of predicting objects and their boundaries from the image. Teams are evaluated with Top-1 or Top-5 accuracy. The higher the Top-1 or Top-5 accuracy, the better the model is.

The classification error of ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dropped exponentially every year as shown in Figure 2.11. In Figure 2.11, the human error rate is quoted 5.1% of the Reference [86]. XRCC of 2011 was the approach of computer vision, and all others are artificial neural networks models. A new CNN-based topology, AlexNet, showed 84.7% accuracy [88], defeating other algorithms that have been studied for decades. At that time, it was a remarkable achievement because it was considered that accuracy of over 80% is impossible.

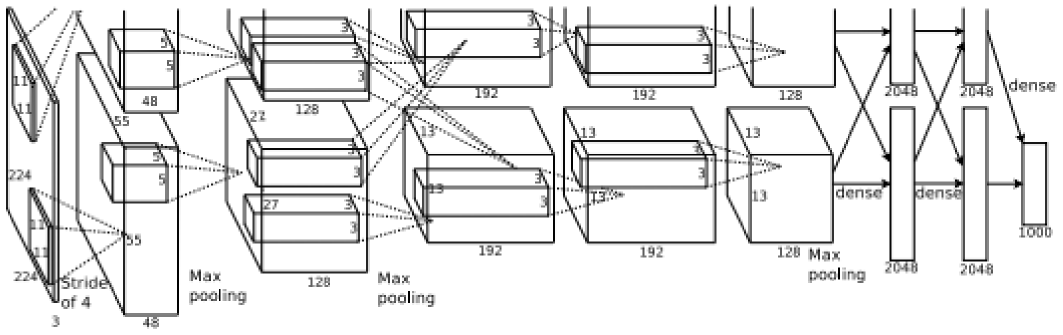


Figure 2.12: Architecture of AlexNet [88]

In 2014, Google reduced Top-5 errors to 6.6% in the ImageNet challenge classification task [89]. In the new model GoogleNet, they proposed a new architecture which is called the Inception module. While existing models have processed data sequentially, GoogleNet can perform operations in parallel. The Inception module helps you to extract fine-grained details to coarse-grained information at once using operations with different filter size.

Microsoft's ResNet is undoubtedly the most significant achievement in the artificial neural networks field over the last few years. Not only has it solved the problems of training

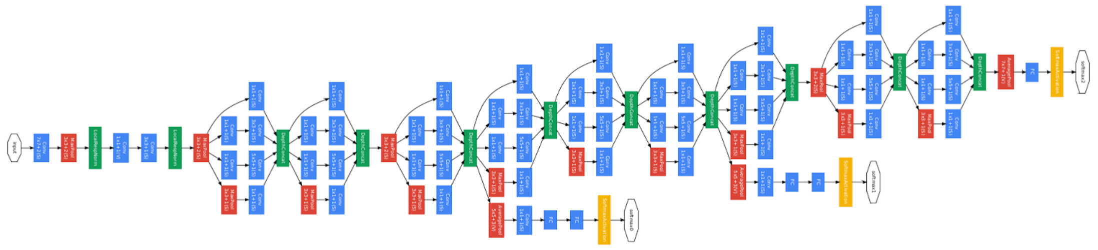


Figure 2.13: Architecture of GoogleNet [90]

deeper neural networks, but also greatly improved the accuracy of the classification task [91]. With the success of ResNet, many computer vision applications like object detection have improved their performance based on ResNet.

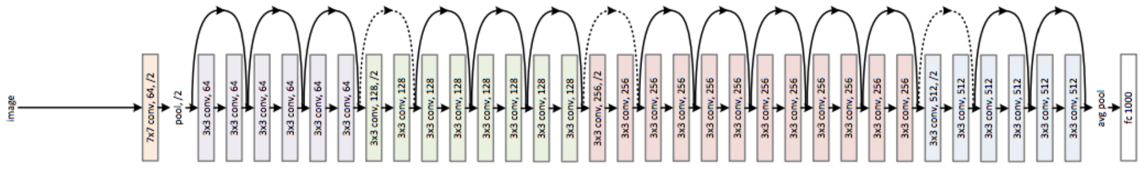


Figure 2.14: Architecture of ResNet [91]

Another interesting event in the field of artificial neural networks was the Google DeepMind Challenge Match. Google Deep Mind's AlphaGo, made of artificial neural networks, won the 18-time world champion Lee Sedol with a score of $4:1$. It's been a while since computers have overwhelmed humans with chess, but it was expected that there would be a different outcome due to a large number of cases in Go. However, AlphaGo became able to count the number of cases at a fast pace since it has trained itself through reinforcement learning based on the combination of advanced hardware technology and artificial neural network technology.

The artificial neural networks technologies that you can directly experience as a user is image recognition. With face recognition from Google or Facebook, you can sort photos by person or tag names in photos. The image recognition algorithm can also search the user's photo album with keywords to find the desired photo. It is a breakthrough in that the machine can analyze the image and link with the language. Even in a recent article by Andrej Karpathy, it has become possible to create a sentence description from an image [92]. They devised a combination of CNN and bidirectional RNN to create a model that generates text descriptions of different regions within an image. It is meaningful that they pioneered a new field with the most actively studied areas in the artificial neural networks community, computer vision and natural language processing.

Artificial intelligence has developed rapidly over the years. There has been no major change in the principles since artificial intelligence was first proposed, but there has been tremendous progress in applying the principles to solve real-world problems. In recent years, artificial neural networks have been used in various fields such as drug discovery, autonomous driving and voice recognition. A series of successes attracted the attention of many researchers, and they again applied various topologies to their research and achieved excellent results. With advances in hardware technology, big data, and the efforts of multiple researchers, the development of artificial neural networks is still ongoing, and the future is more likely.

2.5 Methods

Since traffic flows have both stochastic and nonlinear characteristics, interest in traffic flow prediction using nonparametric methods is increasing [43]. With regard to air traffic, data availability has significantly improved in recent years making a new turning point in measuring and modeling airport capacity. New and improved databases related to airports is now accessible and useable [10]. A good example is the FAA aviation system performance metrics (ASPM) airport data. ASPM airport analysis report provides information on aircraft departure and arrival times and flight delays at selected airports compared to the schedule and flight plan times [70].

This availability of the air traffic data has the potential to improve the fidelity of the model by suggesting future airport capacity models in the direction of data-driven. In this respect, an artificial neural networks approach is a promising method to capture characteristics of throughput of an airport arose from temporal and geographical features.

2.5.1 Multilayer Perceptron

Multi-layer perceptron (MLP) is a system of simply connected nodes arranged in layers [93]. Typical MLP networks consist of three types of layers, an input layer, hidden layers and an output layer as seen in Figure 2.15. The leftmost layer is an input layer, and it has as many nodes as the number of input features. There are two hidden layers in the sample MLP architecture in Figure 2.15 but the number of hidden layers depends on the problem. Input features are passed forward from the input layer to the next layers. The following expression can express the calculation at a hidden node:

$$y_j = f_j\left(\sum_{i=1}^{p} w_{ji}x_i + b_j\right) \tag{2.1}$$

where y_j is the output from the j^{th} node, $f_j(\cdot)$ is the nonlinear activation function of the node and w_{ji} is the weight between the input x_i and the node [93]. At each hidden node, it receives x_i values from the previous layer and performs a simple transfer by calculating the weighted sum. Then the bias value, b_j is added to the weighted sum. It is passed to the next layer after the nonlinear activation is performed [94]. Commonly used activation functions are a sigmoid function and rectified linear unit as shown in Figure 2.16. This nonlinear operation allows you to simulate nonlinearity with a linear weighting model. If MLP can only use linear activation function, it can learn only the linear combination of inputs regardless of how many layers it has.

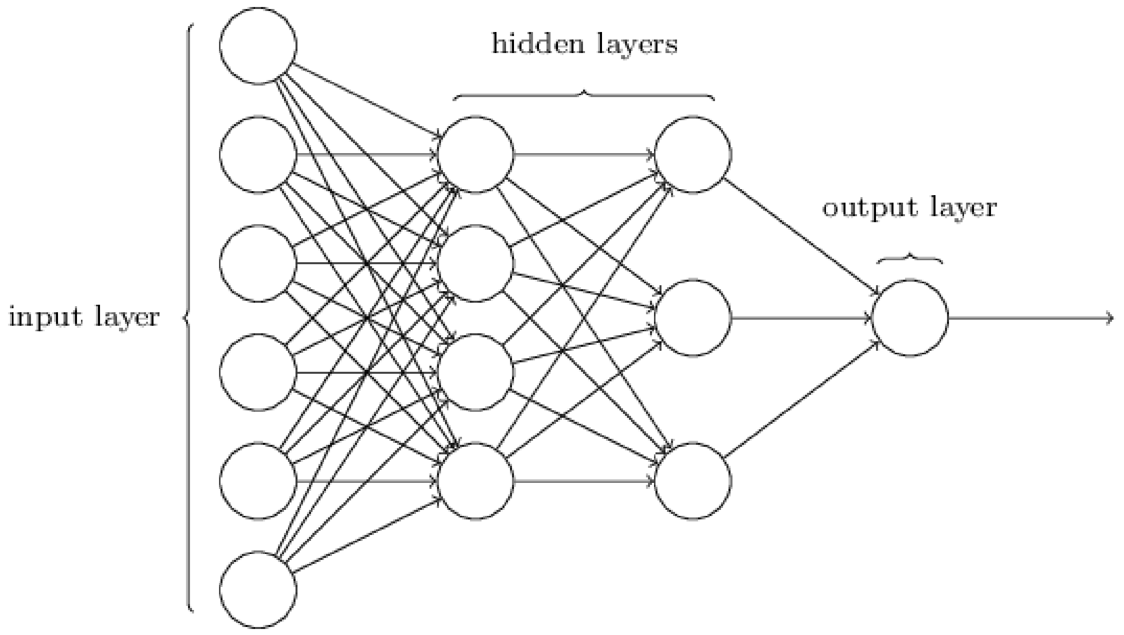


Figure 2.15: Multilayer Perceptron

The rightmost layer in Figure 2.15, the output layer takes the values from the last hidden layer and transforms it into the final result value. For binary classification problems, the output layer yields true/false or 0/1 as results. On the other hand, it will yield a continuous value for a regression problem. Multi-layer perceptron is useful for solving problems stochastically. However, it may not fit well with capacity prediction, where continuity of

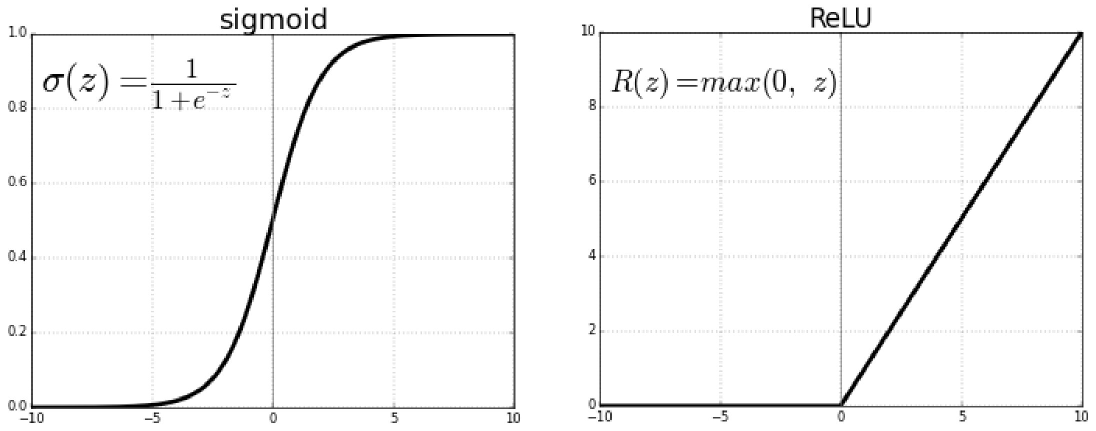


Figure 2.16: Examples of Activation Function

time may be an important factor. Although there is information in the sequence of data itself, MLP tries to infer the results from scratch without considering the information. Because MLP has no notion of order in time.

The back-propagation is the key algorithm of artificial neural networks and is used as a way to learn neurons. When a training sample is fed into the model, the result is obtained at the last layer through feed-forward. Then it calculates the error between the actual value and the output of the model using the loss function selected by the user. The weights of each node are updated by stochastic gradient descent (SGD) method to reduce errors. Conventional gradient descent updates the parameters w to minimize the function $f(w)$ at each iteration:

$$w_{t+1} = w_t - \alpha\frac{1}{n}\sum_{i=1}^{n}\nabla_w f(w_t) \tag{2.2}$$

where α is a scalar chosen adequately. Unlike conventional gradient descent algorithm, SGD computes gradient of only one or a few sample [76]:

$$w_{t+1} = w_t - \alpha \nabla_w f(w_t^i). \tag{2.3}$$

By using reduced number of samples, computational time and memory for training is reduced significantly [95].

2.5.2 Recurrent Neural Networks

Recurrent neural networks (RNNs) also have an input layer, hidden layers and an output layer as the multi-layer perceptron. The difference, however, is that RNNs use the networks to infer an event downstream in time. To do this, RNNs receive sequence of inputs from 1 to T, $(x_1, x_2, ..., x_T)$, and compute the hidden states for that period, $(h_1, h_2, ..., h_T)$. At time t, the RNNs cell takes two sources of inputs, the hidden state of the recent past and the current input as depicted in Figure 2.17. The cell computes a hidden state h_t by applying a linear map to the concatenation of the previous time step's hidden state h_{t1} and the current input x_t. After that, it transfers h_t to the next time step through a nonlinear activation function [96]:

$$\begin{aligned}
a_j(t) &= \sum_{i=1}^{n_I} \alpha_{ji} x_i(t) + \sum_{i=1}^{n_H} \rho_{ji} h_i(t-1) \qquad && j = 1, \dots, n_H \\
h_j(t) &= F(a_j(t)) && j = 1, \dots, n_H \\
b_j(t) &= \sum_{i=1}^{n_H} \beta_{ji} h_i(t), && j = 1, \dots, n_J \\
o_j(t) &= G(b_j(t)), && j = 1, \dots, n_J
\end{aligned}$$

where α_{ji}, β_{ji} and ρ_{ji} are learnable parameters of RNNs. n_H is the number of hidden nodes and is determined by the user. Training examples (x_t, y_t) for $1 \leq t \leq m$ where $x_t \in \mathbb{R}^{n_I}$ is weather and traffic schedule at time step t and $y_t \in \mathbb{R}^{n_J}$ is capacity at time t as the target output. F is an activation function, and G is often chosen as softmax to produce a regression output. RNNs calculate output $o_j(t)$ for $1 \leq t \leq T$.

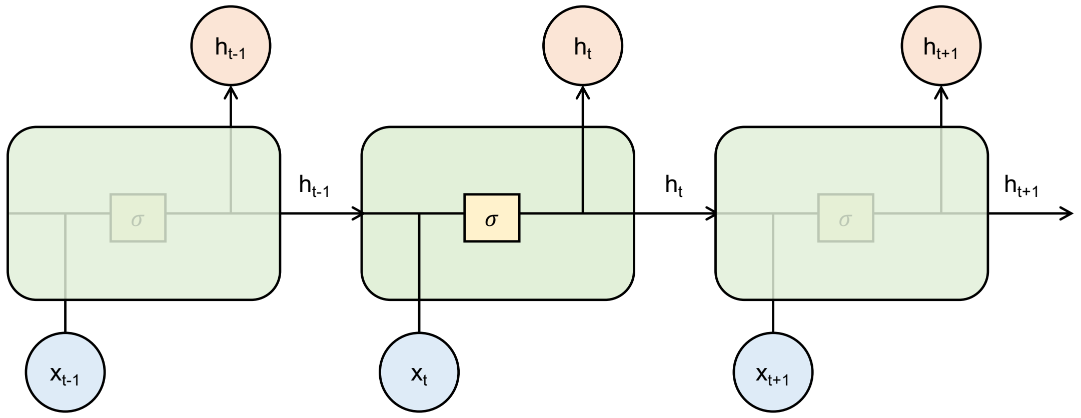


Figure 2.17: Recurrent Neural Network

Unlike MLP, sequential information is preserved in the recurrent neural networks' hidden state. Thus, history in data can be represented by recurrently connected neurons, and RNNs can be trained to compress the entire history into a low-dimensional space [97]. The supervised learning problem of RNNs is defined as predicting the capacity of the current time (t) given the capacity and weather conditions of the previous time step $(t-1)$.

However, there is the problem of long-term dependencies [98]. Predictions of the output at time t can be affected by an input of a long time ago. The problem is that back-propagation, which updates the parameters of RNNs, is not sufficient to detect contingencies over long time intervals [99]. When adverse weather conditions cause a significant drop in airport capacity, the impact may continue throughout the day. So to obtain high accuracy, it would be better to use a method that can discover long-term dependencies.

2.5.3 Gated Recurrent Unit

Recurrent neural networks have shown promising results in many tasks. So multiple variants of RNNs has been introduced to adaptively capture dependencies of different time scales. Among them, the performance of recurrent neural networks with elaborate recursive hidden units, such as Long Short-Term Memory (LSTM) units, is particularly noticeable. From this point of view, Cho et al. proposed a gated recurrent unit (GRU) implementing a sophisticated gating mechanism [100] [101]. Like an LSTM cell, the GRU has a gating unit that adaptively captures dependencies of various time scales, however, without the need for a separate memory cell. Figure 2.18 shows its operating mechanism.

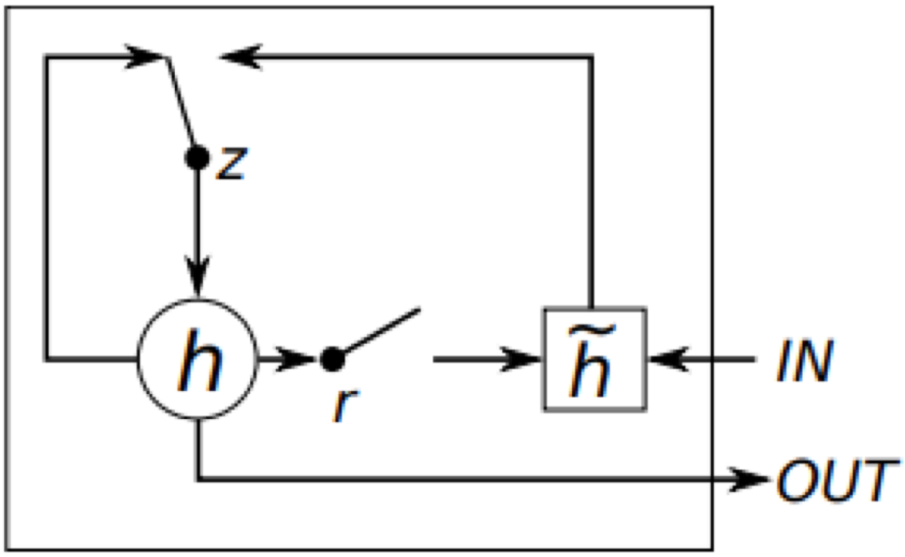


Figure 2.18: Gated Recurrent Unit [101]

The candidate activation of j-th GRU at time t is $\tilde{h}_t^j$ and computed as follows.

$$\tilde{h}_t^j = \tanh(W\mathbf{x}_t + U(\mathbf{r}_t \odot \mathbf{h}_{t-1}))^j$$

where $\mathbf{r}_t$ is a set of reset gates, $\odot$ is an element-wise multiplication and $\mathbf{h}_{t-1}$ is an activation funtion at time $t-1$. The reset gates are calculated as follows and it makes the unit to forget the previously calculated states when its value is 0.

$$r_t^j = \sigma(W_r\mathbf{x}_t + U_r\mathbf{h}_{t-1})^j$$

The activation of GRU, h_t^j is a linear interpolation between the previous time step's activation and the current candidate activation:

$$h_t^j = (1 - z_t^j)h_{t-1}^j + z_t^j\tilde{h}_t^j$$

where z_t^j is an update gate and can be obtained as follows.

$$z_t^j = \sigma(W_z\mathbf{x}_t + U_z\mathbf{h}_{t-1})^j$$

The update gate decides how much the unit update its activation level. The GRU model was built to measure the capacity prediction performance, but the performance did not differ significantly from that of RNN. For that reason, the results were not included.

2.5.4 Long Short-Term Memory

Long short-term memory (LSTM) is a variant of RNN architecture. It is proposed by Hoschreiter and Schmidhuber to cope with the long-term dependencies problem of RNNs [102]. LSTM can learn to store information from extended time intervals [102]. The reason why LSTM can find a correlation between events separated by a long-term is that LSTM has three kinds of multiplication gates: input gate, forget gate, and output gate [98] [103].

An LSTM cell processes the input sequence with three multiplication gates [98] [103]. At time t, the LSTM cell receives the cell state, C_{t-1}, and the hidden state, h_{t-1}, from the previous cell and calculate the current hidden state, h_t and cell state, C_t. The forget gate decides which proportion to forget by receiving the cell state C_{t-1} and the hidden state h_{t-1}

from the previous step:

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + W_{fc}C_{t-1} + b_f).$$

The input gate determines which fraction of the current input is to be transferred to the memory cell:

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + W_{ic}C_{t-1} + b_i).$$

Then the candidate value for cell state at time t is updated:

$$\hat{C}_t = tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c).$$

The new cell state C_t is updated with $\hat{C}_t$ value as follows:

$$C_t = f_t \odot C_{t-1} + i_t \odot \hat{C}_t$$

where $\odot$ is the component-wise product. With the new cell state C_t, the output gate updates the current hidden state, h_t, for passing it to the next step:

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + W_{oc}C_t + b_o)$$

$$h_t = o_t \odot tanh(C_t).$$

2.5.5 Methods for Better Convergence and Fast Training

Stochastic Gradient Descent (SGD) [95] [104]

Firstly, let us consider simple supervised learning. Suppose that models trained by supervised learning are parameterized by function $f_w(x)$ of the weight vector w. Each example used to train a model consists of a scalar output y, and a feature vector **x** of schedule and

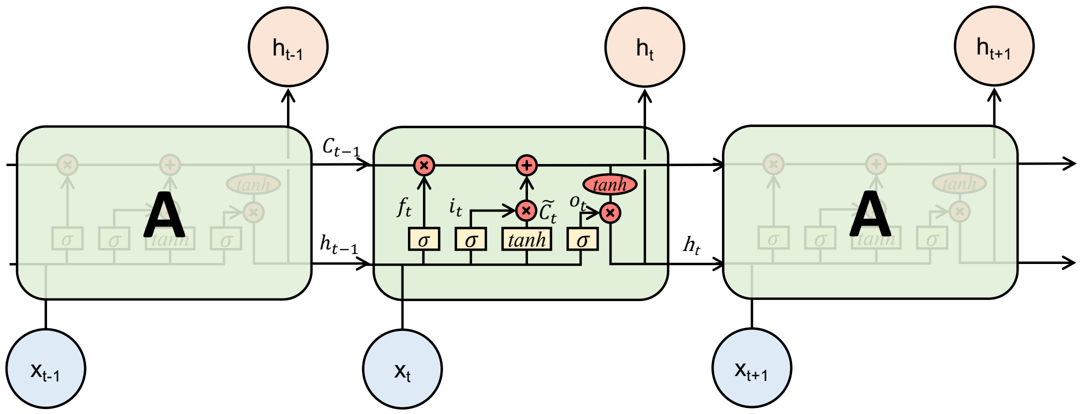


Figure 2.19: Long Short-Term Memory

weather at the airport. The loss function $\ell(\hat{y}, y)$ measures the cost when the actual output is y and the model's prediction output is $\hat{y}$. The training of artificial neural networks models aims to find a function $f_w(x)$ that minimizes the averaged loss $Q(z, w) = \ell(f_w(x), y)$ on training samples z.

Rumelhart et al. proposed the gradient descent algorithm to update the weights w every iteration with the following equation [82].

$$w_{t+1} = w_t - \eta \frac{1}{n} \sum_{i=1}^{n} \nabla_w Q(z_i, w_t)$$

η is an appropriately selected learning rate that is neither too large nor too small. It has been mathematically proved that it can converge to optimum with randomly initialized w and sufficiently small learning rate η.

If there is a large number of training samples, using the gradient descent algorithm against the entire sample may take too long. To compensate for this, stochastic gradient descent (SGD) algorithm has been proposed, which is a simplified version of the gradient descent algorithm. Instead of averaging the gradients of all samples $\nabla_w Q(z_i, w_t)$ for $i = 1$ to n, SGD computes the gradients only for the randomly selected subset at each step. The pseudocode of SGD is described in the Algorithm 1. The distribution of classes within each subset should be kept uniform so that the gradient is almost the same in any subset.

Algorithm 1 Stochastic Gradient Descent Algorithm

```
Choose learning rate $\eta$ and subset size $b$
Initialize weights $w$
while Not converged do
    Shuffle $n$ training examples
    for i=1, $\ldots, \frac{n}{b}$ do
        Pick $b$ examples randomly from the training set
        $w_{t+1} = w_t - \eta \nabla_w Q(z_t, w_t)$
    end for
end while
return $w$
```

Learning Rate Decay [104]

A learning rate is an important parameter in SGD. The learning rate of SGD determines the step size of the parameter update in each iteration. If this parameter is too large, the training algorithm can diverge. In the opposite case, the algorithm may converge too slowly. It is helpful in almost every case to initially set the learning rate to a reasonable size and then reduce it later. This has the effect of reducing random variations in errors due to gradient differences between different iteration. It is also a good idea to decay learning rate when there is a point where the error value no longer decreases.

Starting with a certain learning rate and gradually decreasing the value by a rule has already been empirically proven by many studies and is called 'learning rate decay'. Learning rate decay is a kind of simulated annealing to approximate global optimization within the search space [105]. Learning rate is decayed in a way that balances exploration and exploitation. It is to explore large areas in the early stages and look closely at the areas of confidence in the latter part. The large learning rate value at the beginning of training finds the optimal region with the possibility of a global minimum. As the training progresses, the learning rate is decreased, and the minimum within the area is found. According to previous research by Robbins and Monro, the learning rate should be decayed to satisfy the following conditions to find the minimum [104].

$$\sum_{i=1}^{\infty} \eta_i = \infty$$

$$\sum_{i=1}^{\infty} \eta_i^2 < \infty$$

Examples of learning rate decay schedules include exponential decay, cosine decay [106], and step-function drop [107].

Momentum SGD

SGD suggests a fundamental way to find the optimum, but it tends to be slower due to oscillation near optimum. Momentum helps to reduce swings in SGD. Momentum SGD accelerates SGD [108] by adding a term called momentum to the equation that updates the weights [82].

$$v_{t+1} = \gamma v_t - \eta \frac{1}{n} \sum_{i=1}^{n} \nabla_w Q(z_i, w_t)$$

$$w_{t+1} = w_t - v_t$$

In the above equation, γ is the momentum parameter. As this term is added, the weights in the next step are determined by the current weights vector along with the current gradient [109].

In particular, the convergence rate of SGD is slow when there are long and narrow valleys on the $Q(z, w)$ function surface. SGD oscillates along the short axis because the gradient points to the valley wall, not the center of the valley. The momentum term stabilizes the gradient that oscillates toward the wall using the weight of the previous value. In addition, it can add contributions to the weights when the gradient points to the minimum [109].

Adaptive Moment Estimation (ADAM) [110]

Adaptive Moment Estimation (ADAM) is an algorithm for first-order gradient-based optimization with little memory requirement. This method calculates the adaptive learning rate by estimating moments of the gradient. The first moment m_t and the second moment v_t are defined as follows:

$$g_t = \nabla_w Q$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

where β_1 and β_2 are hyperparameters to control the exponential decay rates of the moment estimates. g_t^2 is the elementwise matrix multiplication, $g_t \odot g_t$. Since m_t and v_t are initialized to zero vectors and have an biased value to zero, the corrected estimate $\hat{m}_t$ and $\hat{v}_t$ are computed.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

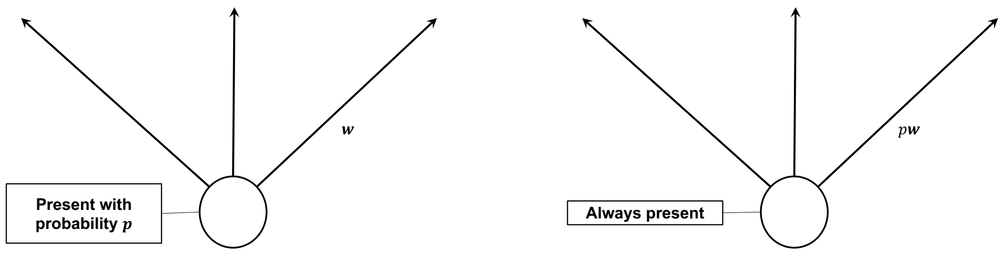
Then the bias-corrected moments are used to update the weights as follows:

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

where ϵ is another hyperparameter whose default value is 10^{-8}.

Dropout [111]

Deep neural networks can model complex systems with many parameters. However, overfitting often occurs when the number of model parameters and layers are too large than the size of data. Dropout is a technique that resolves the overfitting problem by dropping nodes and their connections randomly during training in the neural networks. Each node is a retained at a fixed probability of p, and the retention of the node is determined independently of the other nodes. The test phase uses a single neural network with no dropout. The difference from is that if the nodes are retained with the probability p during training, the outgoing weight of the nodes is multiplied by p at test phase.



(a) At Training (b) At Test

Figure 2.20: Schematics of Dropout at Training and Test Phase [111]

2.6 Consideration of the Markov model [112]

In addition to the recurrent neural network, there is also a Markov model that can represent time dependencies in the data. Markov models are frequently used in predictability studies. A Markov chain is the simplest Markov model proposed by Andrey Markov to model the transition between probabilistic states in the observed order under the Markov assumption. The Markov model assumes that the distribution of the state at time t depends only on the state of the previous time $t - 1$ [113]. Moreover, Hidden Markov models (HMMs) is a Markov chain in which states are only partially observed. Hidden Markov models (HMMs) can model the system of a partially observed sequence as a probability model [114]. Strictly speaking, HMMs can infer the hidden state from the observed output under the same assumptions as the Markov chain.

However, these Markov models are limited in their use because they must draw their state from a discrete state space S of a reasonable size. Then, the size of the transition table capturing the probability that any two adjacent states transition is $|S|^2$. Therefore, as the number of possible hidden states increases, it becomes difficult to execute the operations required for HMMs. In this respect, the main advantage of RNNs compared to Markov models is the greater representational power. This advantage is more noticeable when the size of the context window that the model predicts is large. Theoretically, it is not impossible to model long-range dependencies extending the Markov model. However, extending the Markov model according to the size of the context window and the number of possible hidden states exponentially increases the number of possible transitions between states [115]. Therefore, applying the Markov model to the capacity prediction problem, which has a large number of possible states in the broad context window, makes the calculation impractical.

Another limitation of applying Markov models to the capacity prediction problem arises from the assumption of independence. The Markov assumption states that the future is independent of the past given the present [116]. Accordingly, Markov models try to find the current state from the previous discrete state. However, this is not always true in real life. The information needed to predict the current state can sometimes be in a state a few time steps back. In the capacity prediction problem, the current value may vary depending on the capacity of the steps before the previous step even if the capacity of the previous stage is the same. Unlike the Markov model, RNNs can learn and infer long-range time dependency by a combination of the current input and the states of the hidden layers.

2.7 Data Source and Processing

The models are trained using hourly capacity and weather data from 2013 to 2017 at one of the busiest airports, HartsfieldJackson Atlanta International Airport (ATL). Since ATL airport underwent major changes in 2012 due to the expansion of international terminals, data from 2013 onwards were used. The hourly capacity data for ATL was obtained from the airport analysis module of the Federal Aviation Administration (FAA)'s Aviation System Performance Metrics (ASPM) database [117].

The ASPM is an online system that provides data on flights operated by major airlines departing from or arriving at the major 77 airports. The airport analysis module provides information on aircraft departure and arrival rates for selected airports. It also includes information about hourly flight delays at the airport compared to flight schedules. Note that this data excludes general aviation, military traffic, local(non-itinerant) traffic and international flights [117]. The ASPM data is updated based on Airline Service Quality Performance (ASQP) data. ASQP data includes OOOI (Gate Out, Wheels Off, Wheels On, and Gate In, or Out-Off-On-In), final schedule data, and the cause of delay reported by airlines. From the ASPM database, the scheduled/actual number of departures and arrivals in a defined period are extracted.

- Month

- Day of Month
- Local Time
- Actual Departure Rate
- Actual Arrival Rate
- Scheduled Departure Rate
- Scheduled Arrival Rate

The weather data is included in the training data because it is evident that the airport capacity changes in response to weather conditions such as visibility and ceiling. The weather conditions determine whether the aircraft will follow visual flight rules (VFR) or instrument flight rule (IFR). In addition, the separation distance between aircraft is affected by this, so the number of aircraft that the airport can serve per hour varies by weather [72]. The terminal weather data for each hour was collected from the National Oceanic and Atmospheric Administration (NOAA)'s Integrated Surface Database (ISD) [118]. The Integrated Surface Database (ISD) provides observations from a number of sources into one common ASCII format. The database is updated daily from more than 14,000 active stations worldwide. It includes various fields measured at each station, including wind speed and direction, temperature, cloud height, sea level pressure, visibility, rainfall, snow depth. The ISD archive data can be accessed in a variety of ways. In this study, FTP access was used to obtain a large amount of data efficiently.

- Wind Direction Angle [deg]
- Wind Speed [m/s]
- Cloud Height [m]
- Visibility [m]

- Temperature [°C]
- Pressure [hPa]
- Precipitation [mm]
- Snow [cm]

The capacity and the weather data tables are joined using the date and time as the joining keys. That way, each row of the joined table consists of capacity per hour and the weather at that time. Then, the merged dataset is subjected to a preprocessing step. By preprocessing, all of the categorical variables are converted to a one-hot vector since artificial neural networks algorithms exhibit better performance with numerical variables. Linear interpolation is used to deal with missing values of weather data using two adjacent known values. Furthermore, features are normalized and the range of features are scaled to prevent the model from being dominated by a few features that have a high variance. In the end, this dataset is split into training data and test data.

2.8 Experiments

Formally, the hourly capacity prediction is to model the posterior distribution $\rho(y_t|Y_{1:t-1}, X; \Theta)$ of the capacity y at time t, given air traffic schedule and terminal weather X_t and the capacity history $Y_{1:t-1}$. Training samples, (x_t, y_t), for $1 \leq t \leq T$ consist of features, including weather and traffic schedules, and a corresponding response. Each feature, x_t^i, that constitutes a single sample represents either the terminal's weather or airport flight schedule. On the other hand, $y_t \in \mathbb{R}^{n_J}$ is the actual capacity at time t used as the target output. The capacity in the previous state is used to predict the current capacity along with terminal weather and flight schedules. The models trained with MLP, RNN and LSTM are parameterized by Θ.

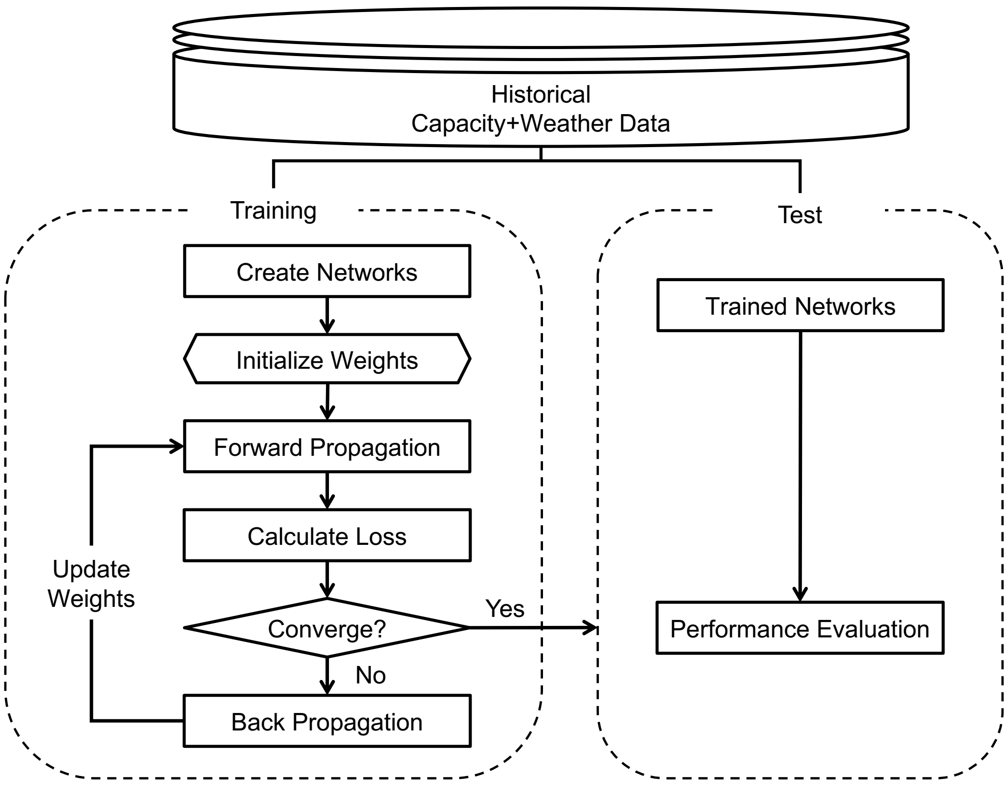


Figure 2.21: Flow Chart for Hourly Capacity Prediction

The overall process for developing a capacity prediction model is illustrated in Figure 2.21. Historical capacity and weather data are collected from the database, respectively, and merged. Then the merged dataset is randomly divided into training data and test data. MLP, RNN, and LSTM are constructed by stacking the building blocks that make up each model. MLP uses a fully connected layer and activation as a building block, and the building blocks of RNN and LSTM are the RNN cell and the LSTM cell described in Section 2.5. Then the parameters of the networks are initialized using Xavier algorithm. The training dataset is forward propagated to the networks. The networks produce output from the calculation with the model parameters. From this, the loss function is calculated using the difference between the actual and output values of the model. If the condition for convergence of the model is not satisfied, the error value will be back-propagated through the network. Back-propagation updates model parameters and the iteration continues until the model converges. For the trained model, the performance evaluation is performed using the test data.

Table 2.2: Structures of MLP, RNN, and LSTM

	Number of Hidden Layers	Number of Hidden Units
MLP	2	{65, 30}
RNN	2	{200, 200}
LSTM	2	{250, 250}

The performance of MLP, RNN, and LSTM created through the process of Figure 2.21 depends heavily on how the layers are constructed. The network structure of MLT, RNN, and LSTM were determined by the grid search method, and the number of hidden layers and the number of hidden units are shown in Table 2.2. As shown in Figure 2.22, MLP is provided with a subset of the training set every iteration via the input layer. In the case of MLP, the airport capacity at a particular time and the weather features at that time

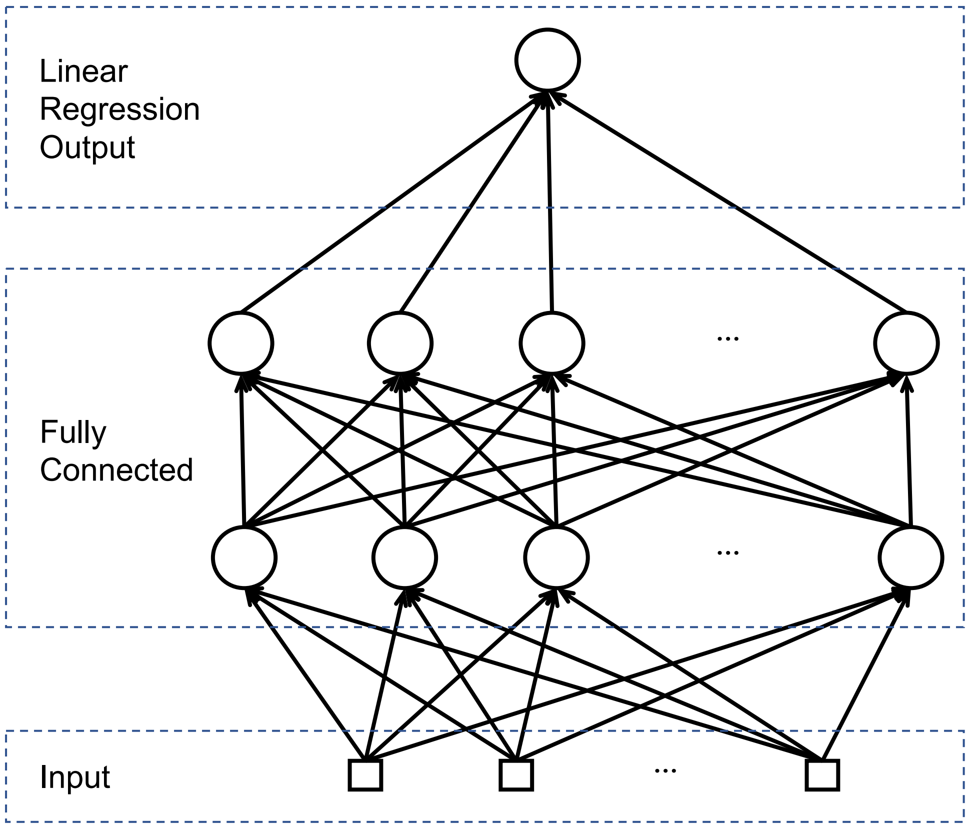


Figure 2.22: Architecture of MLP

constitute one sample. After the input layer, there are two hidden layers, each with 65 and 30 hidden nodes followed by a *tanh* activation function. The result of the last hidden layer is combined to yield the float result in the linear regression output layer.

On the other hand, RNN and LSTM in Figure 2.23 use the hourly airport capacity of k hours as the input sequence. The 2-layer RNN with 200 hidden nodes has been created, and each RNN layer has a dropout regularization with a probability of 0.3. Dropout is a technique that prevents overfitting. While training the network, it retains the node with

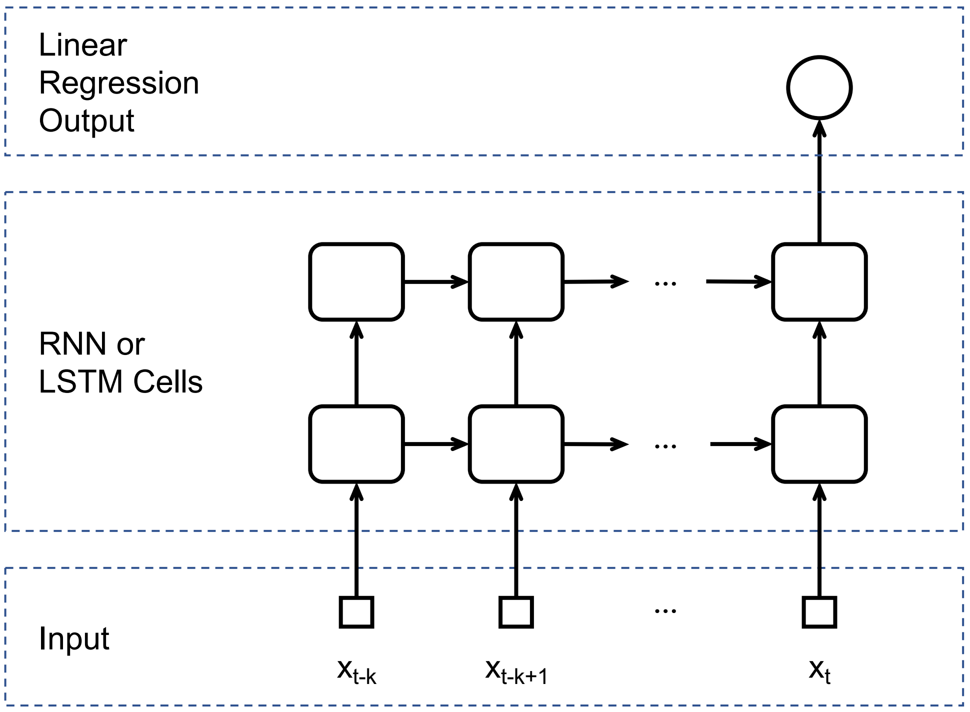


Figure 2.23: Architecture of RNN and LSTM

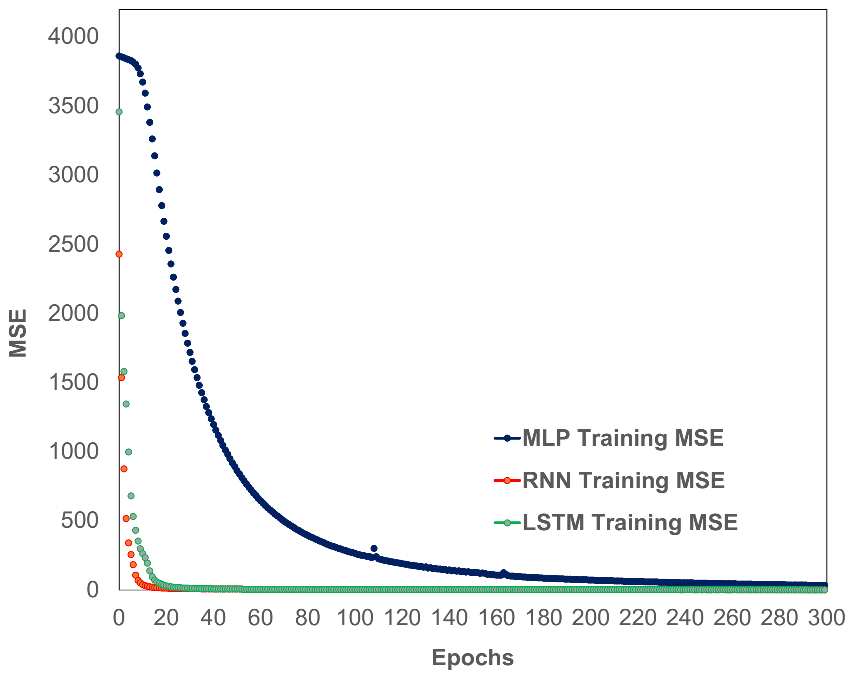
a probability of p and deletes connection between the nodes with a probability of $1 - p$. The LSTM model consists of two layers and 250 hidden nodes per each layer. After each LSTM layer, dropout is applied with 30% probability, like RNN. RNN and LSTM contain a linear regression output layer after the last depth of the structure shown in Table 2.2. In this way, it is possible to obtain an output value for the regression.

Several combinations of layers have been tested to find the best model, but increasing the number of hidden nodes or stacking more hidden layers did not significantly improve performance beyond the model shown in the Table 2.2. Increasing the number of layers or the number of hidden nodes per layer may slightly improve training accuracy. However, as the number of parameters to train increases, the amount of computation grows and the possibility of overfitting increases.

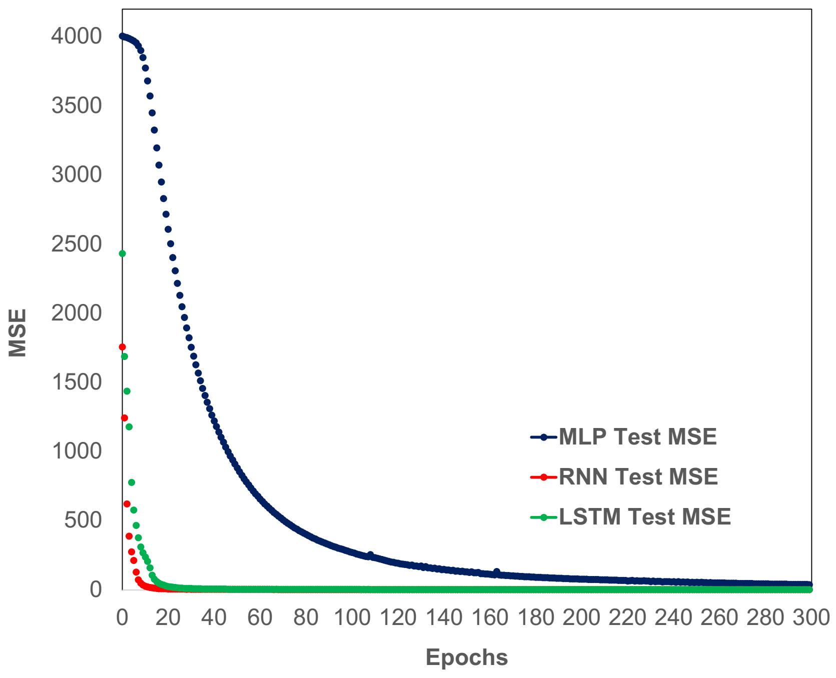
All three models are implemented using MXNet 1.0.0 [119], the open source library for deep learning in Python 2.7.14 and GCC 4.2.1 environments. The latest version of MXNet is available at https://mxnet.apache.org/.

2.9 Results

MLP was trained by stochastic gradient descent algorithm whereas RNN and LSTM were trained by Adam algorithm, an algorithm for the first-order gradient-based optimization algorithm. The models were trained for 300 epochs, and exponential learning rate decay was applied to prevent the model from overshooting the minima. Figure 2.24 and Figure 2.25 show how the training MSE and the test MSE of each model change over 300 epochs. All three models well converged with relatively small MSE values. MLP's training MSE and test MSE had steadily decreased while the curves of the other two models dropped sharply in the early stages and soon became flat. Comparing the training MSE and the test MSE, no signs of overfitting was found in all three models.

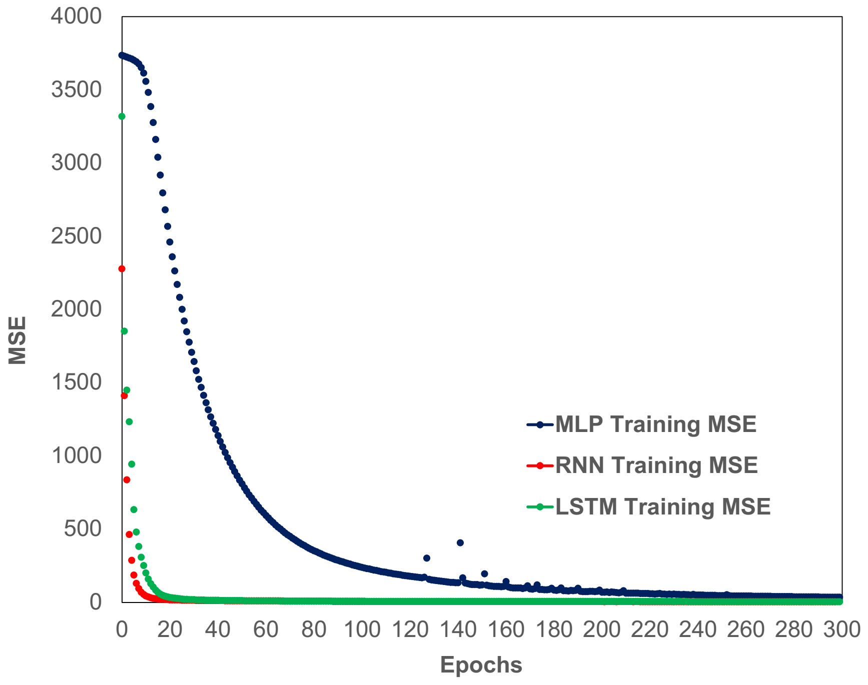


(a) Training MSE

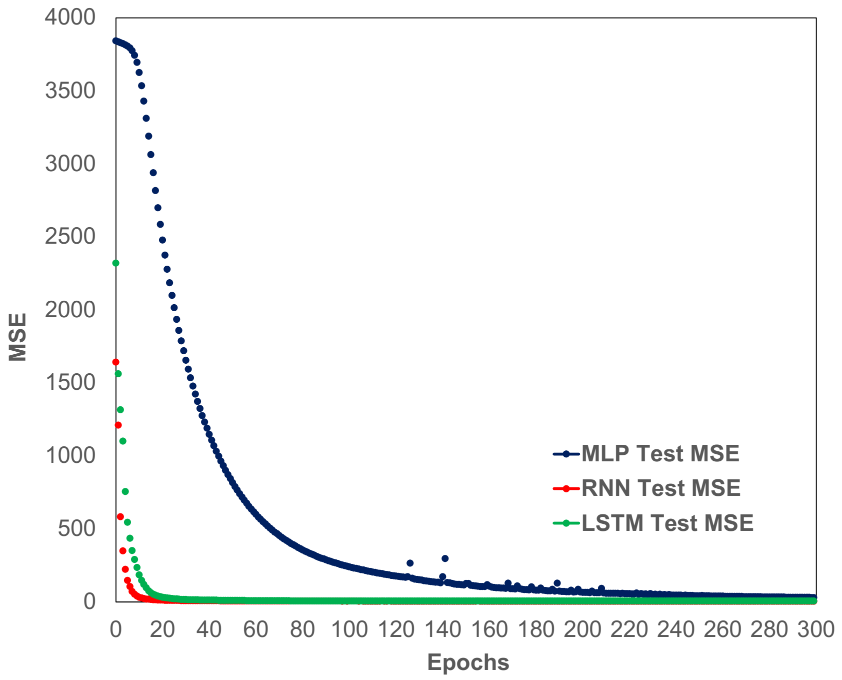


(b) Test MSE

Figure 2.24: Models' Training History for Departure Capacity



(a) Training MSE



(b) Test MSE

Figure 2.25: Models' Training History for Arrival Capacity

2.9.1 Actual vs Predicted

From 2.26 to Figure 2.31 are group of scatter plots comparing 3 models' departure and arrival capacity prediction and actual hourly departure capacity to analyze the performance of three models qualitatively. The vertical and horizontal axes are the predicted and the actual capacity, respectively. The diagonal line in the figures indicates the state where the error is zero since the predicted value and the actual value match. Thus, the more significant the difference between the actual and the predicted value, the more the points deviate from the diagonal. In predicted versus actual plot, LSTM and RNN showed better performance than MLP. MLP underpredicts large capacity values so that the points are below the diagonal at the top right of Figure 2.26 and Figure 2.29. RNN and LSTM well predicted the departure and the arrival capacity of all ranges, so the points in Figure 2.27, Figure 2.28, Figure 2.30 and Figure 2.31 make diagonal trend and the number of outliers is small. The advantage of being able to acquire knowledge from the data sequence seems to be beneficial.

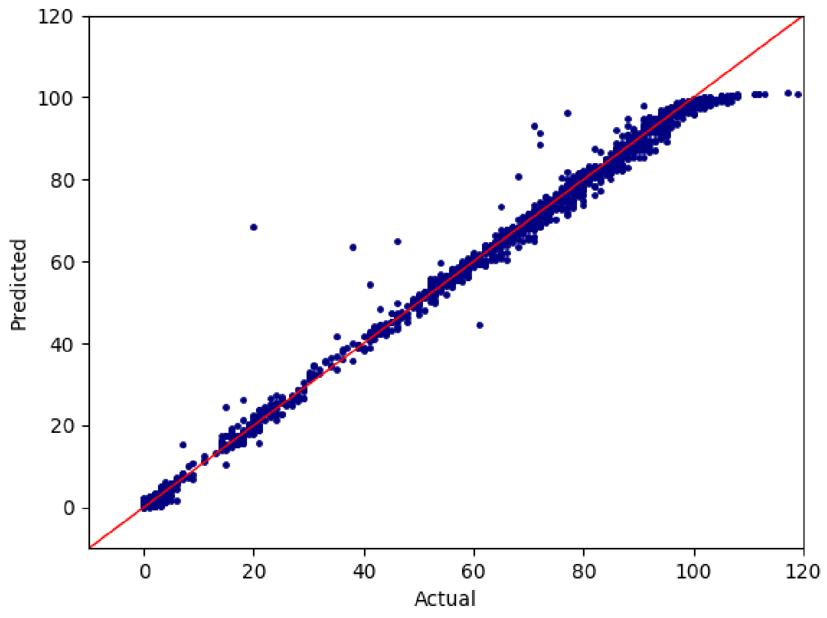


Figure 2.26: Predicted versus Actual Plot of MLP for the Hourly Departure Capacity of ATL

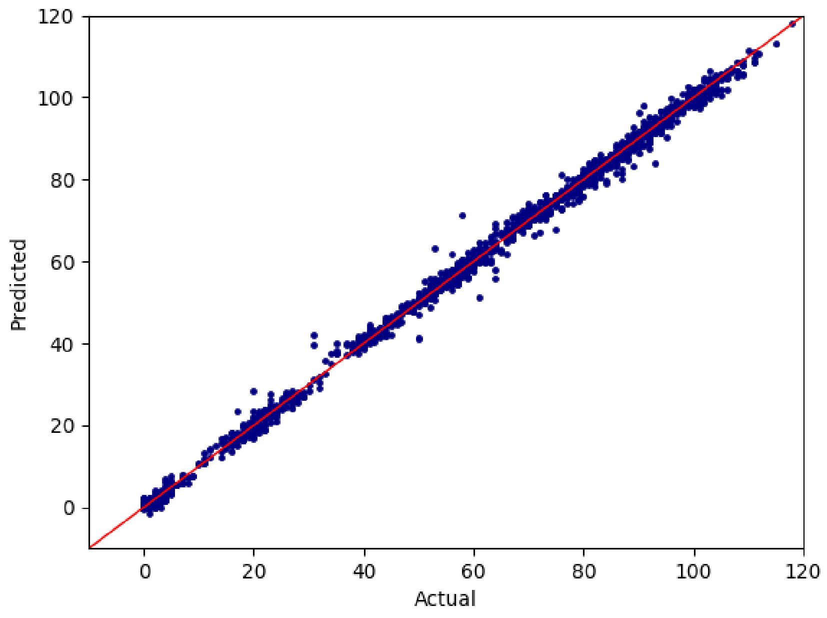


Figure 2.27: Predicted versus Actual Plot of RNN for the Hourly Departure Capacity of ATL

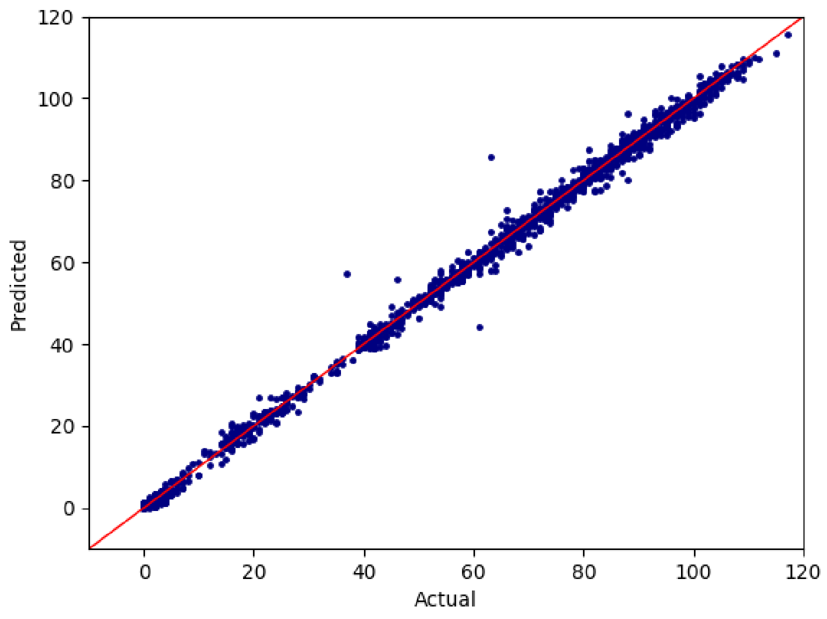


Figure 2.28: Predicted versus Actual Plot of LSTM for the Hourly Departure Capacity of ATL

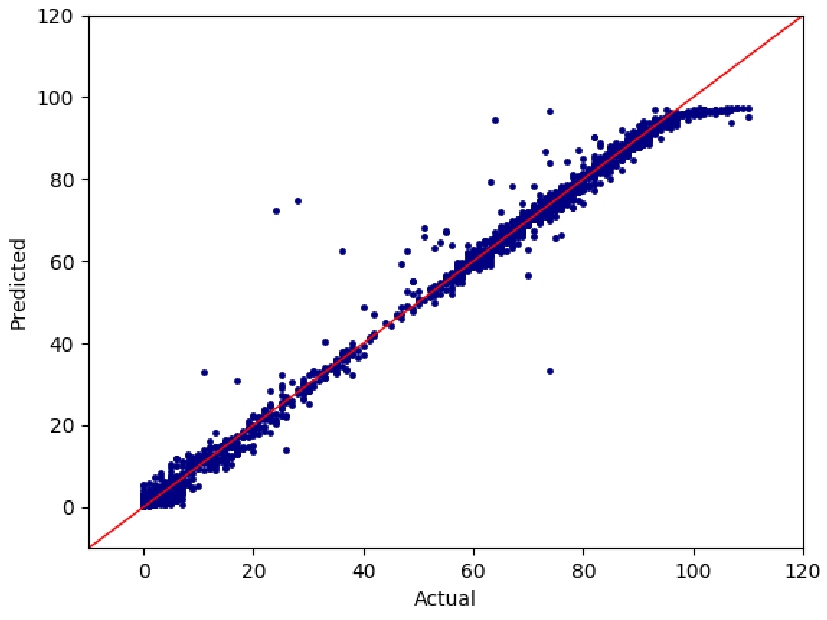


Figure 2.29: Predicted versus Actual Plot of MLP for the Hourly Arrival Capacity of ATL

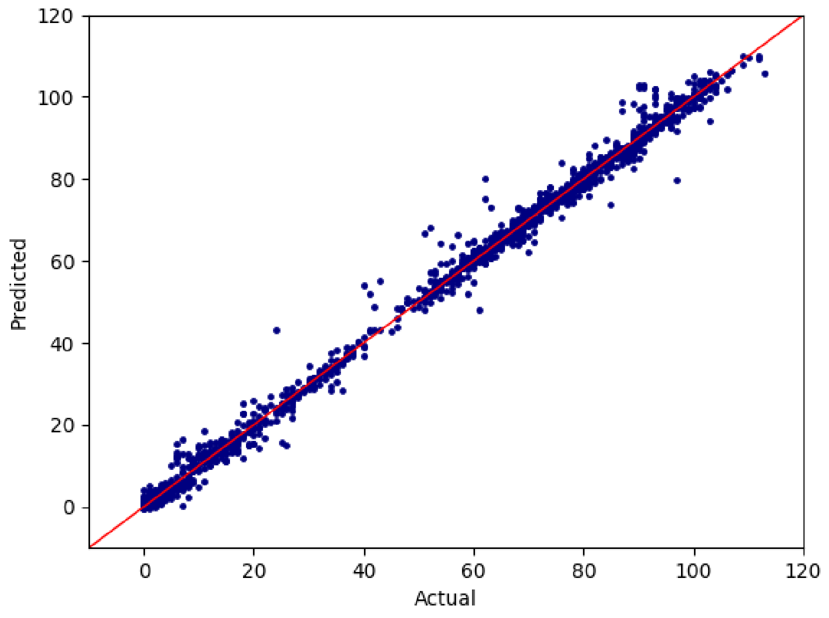


Figure 2.30: Predicted versus Actual Plot of RNN for the Hourly Arrival Capacity of ATL

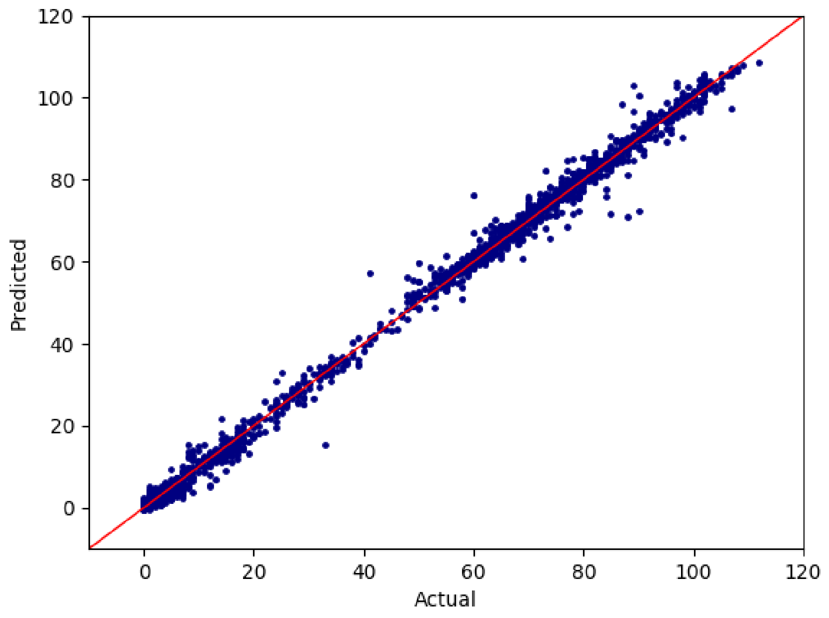


Figure 2.31: Predicted versus Actual Plot of LSTMfor the Hourly Arrival Capacity of ATL

2.9.2 RMSE Comparison

In Table 2.3 and Table 2.4, the best RMSE found in each model after training were compared. All of the proposed approaches in this paper yielded low RMSEs. In the hourly departure capacity forecasting in Table 2.3, even the simplest model, MLP, could show a decent RMSE. The training error of LSTM was the lowest in both departure and arrival, followed by RNN and MLP. Looking at the error trends of the three models, the departure RMSE was lower than that of arrival. Since the models better predict the departure capacity, it may be easier to extract the information related to the departure capacity from the input features than the arrival capacity. The predictability that the FAA and Eurocontrol jointly investigate supports this. The level of variability measures predictability in each flight phase and departure time variability was lower in the United States [120]. This means better punctuality of departure than arrival and better predictability of departure.

Table 2.3: RMSEs of Hourly Departure Capacity Prediction for ATL

	MLP	RNN	LSTM
Training RMSE	3.22	1.44	1.31
Test RMSE	2.98	1.80	1.77

Table 2.4: RMSEs of Hourly Arrival Capacity Prediction for ATL

	MLP	RNN	LSTM
Training RMSE	3.69	2.07	1.30
Test RMSE	4.27	2.51	1.84

2.9.3 Generalizability of Models

Figure 2.32 and Figure 2.33 show the test results of how well the model predicts data that it has never seen before. The capacity dataset for January 2018 was held back from model training and used to test the performance of models trained with data from 2016 to 2017. Model's performance was assessed by the cumulative distribution function of the relative error between the predicted and the actual using the validation set. The cumulative distribution function makes it easy to see the percentage of samples that the relative error is less than or equal to a specific value. Although RNN and LSTM are more complex models that have more trainable parameters than MLP, the ability to predict departure and arrival capacity was lower than MLP. The simplest model, MLP, predicted more than 80% of the departure and arrival capacity samples with the relative error less than 0.1.

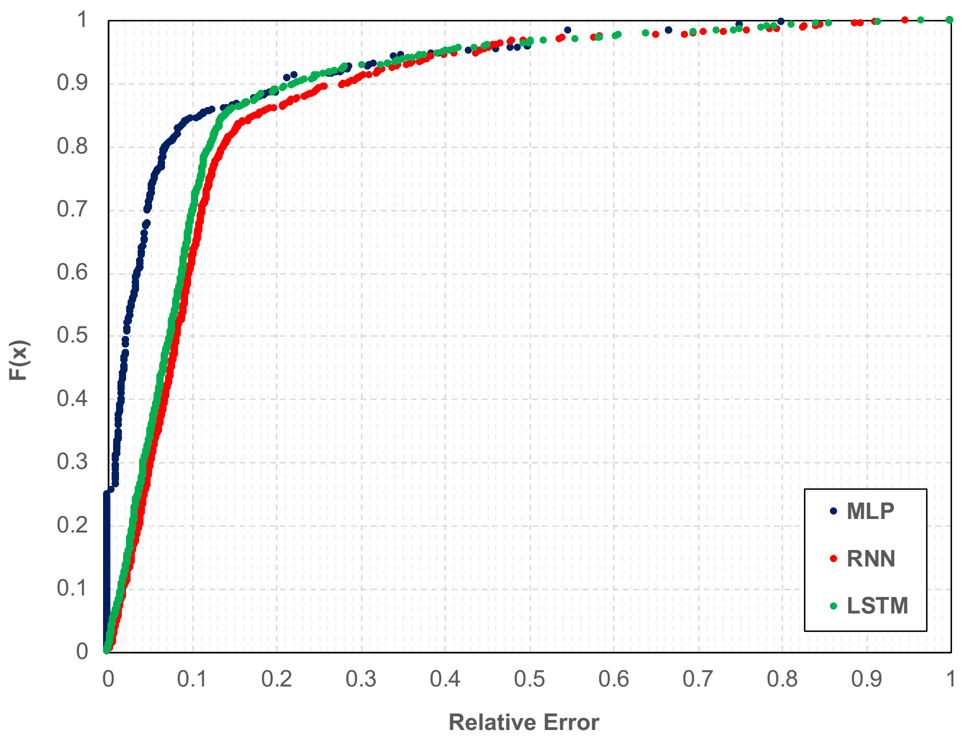


Figure 2.32: Cumulative Distribution Function of Relative Error for ATL Departure Capacity in January 2018

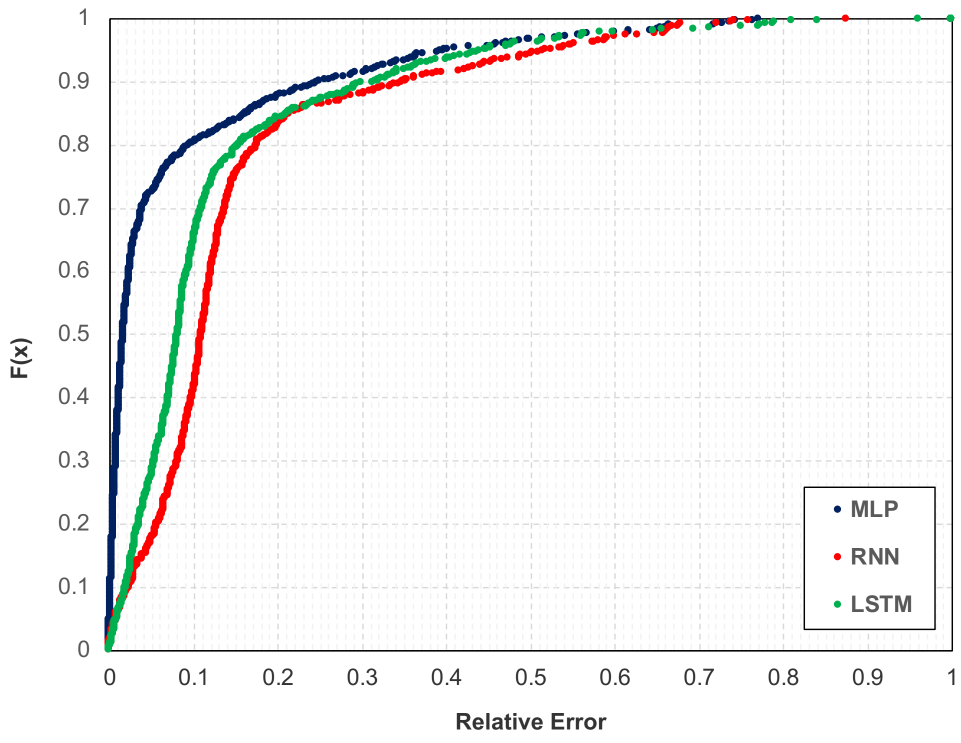
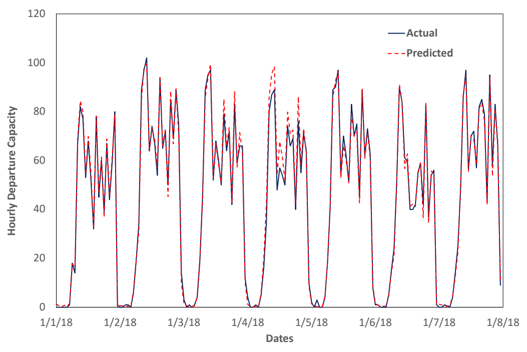
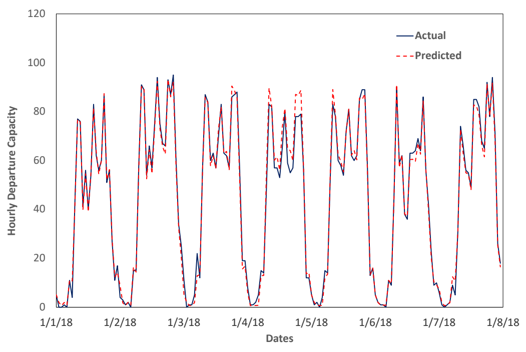


Figure 2.33: Cumulative Distribution Function of Relative Error for ATL Arrival Capacity in January 2018

In order to compare the practical prediction ability, the actual and predicted capacity from January 1^{st} to 7^{th}, 2018 are shown in Figure 2.34, Figure 2.35 and Figure 2.36. MLP had small gaps in the peak time of January 4^{th}, 2018 but predicted both departure and arrival capacity reasonably well. The graphs of RNN and LSTM followed the increasing and decreasing trends of actual values but lacked the ability to predict correct values. LSTM tended to predict the capacity of peak hours to a certain value higher than the actual value. This result can be interpreted as the fact that the data context window was short enough that MLP could perform complex modeling. It also means that tuning the parameters of RNN and LSTM is trickier compared to MLP. There is room for performance improvement for these models.

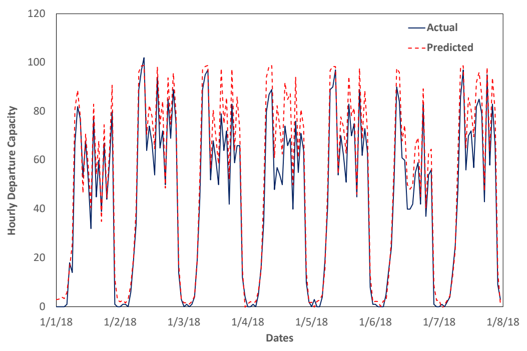


(a) Departure

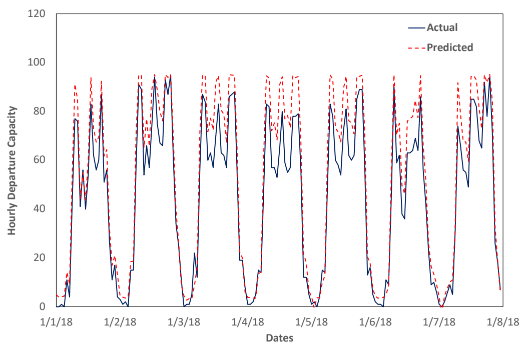


(b) Arrival

Figure 2.34: MLP Prediction for ATL Capacity in January 1^{st} to 7^{th} 2018



(a) Departure

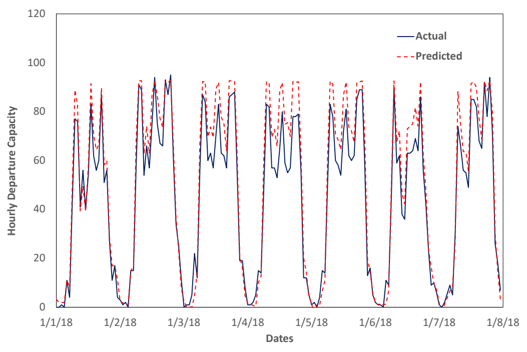


(b) Arrival

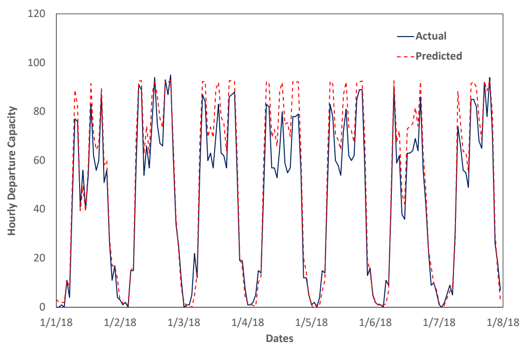
Figure 2.35: RNN Prediction for ATL Capacity in January 1^{st} to 7^{th} 2018

2.9.4 Transfer Learning

This time, the experiments were carried out to see if the model was able to use the knowledge gained from ATL to other airports. The idea of reusing pre-trained models is actively being studied in the field of artificial neural networks. This technique is called *transfer learning*. Transfer learning is to apply a pre-trained model to a new job or a new domain [121]. The transfer of artificial neural networks can speed up training on the target problem using the parameters of the networks trained for the related source tasks [122]. Another advantage of transfer learning is that it can prevent overfitting. Overfitting can occur when



(a) Departure



(b) Arrival

Figure 2.36: LSTM Prediction for ATL Capacity in January 1^{st} to 7^{th} 2018

training a large model with a small number of data. With transfer learning, the smaller target data than the source data can train the model without overfitting [123]. In most real-life problems, transfer learning is useful because it is difficult to find large and well-labeled data. If transfer learning can be applied to the models created in this study, it may not be necessary to train the model for each airport separately from scratch. It also means that one may efficiently generate models for each airport by tuning existing models with a small amount of data.

To demonstrate the transferability of the predictive models, the ATL models were reused to predict the capacity of Boston Logan International Airport (BOS). The process for transfer learning is illustrated in Figure 2.37. The color code in Figure 2.37 distinguishes the source and the target. Dark blue means ATL, so Figure (a) means ATL data and the source networks trained with it. The objects marked in red are the data of BOS and the target networks trained with it. The RNN or LSTM layers in Figure 2.37(b) were represented by red and blue checkered in that the weights are taken from the ATL networks in Figure 2.37(a) and fine-tuned with the BOS data. The last fully connected (FC) layer of the source network is replaced with a new layer. Unlike the transferred layers, this layer is randomly initialized and trained with BOS data.

The transfer learning method described in Figure 2.37 does not apply to MLP. The rea-

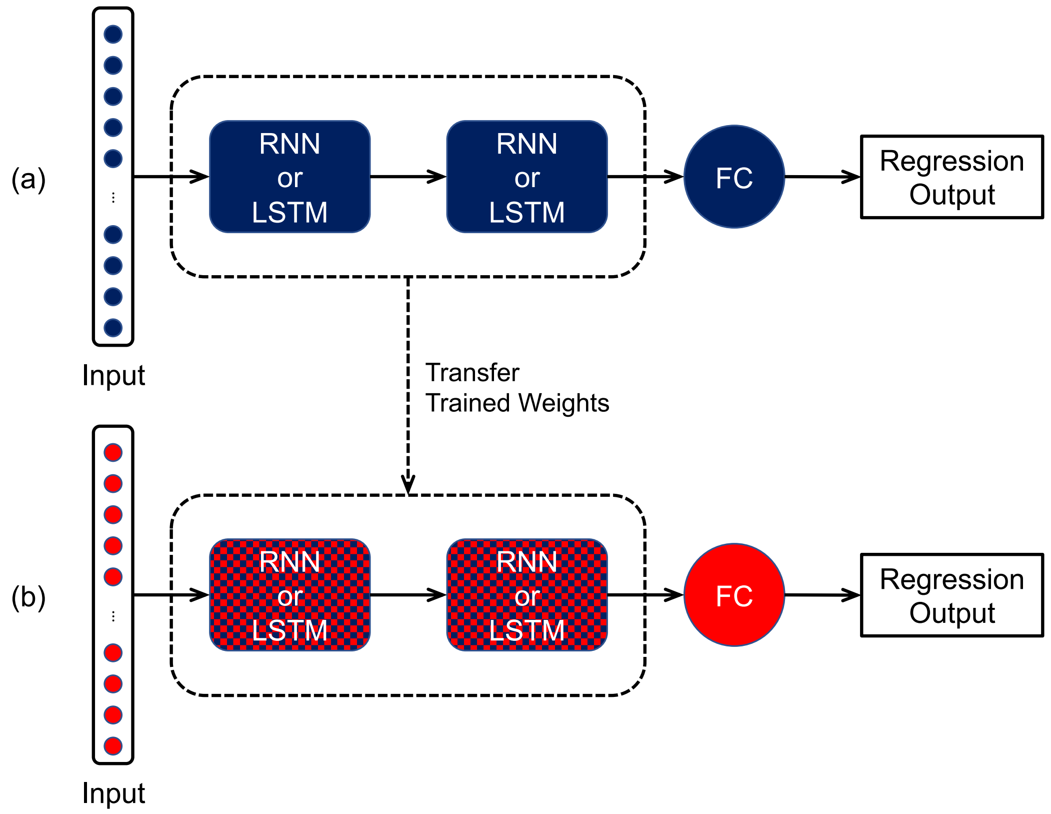
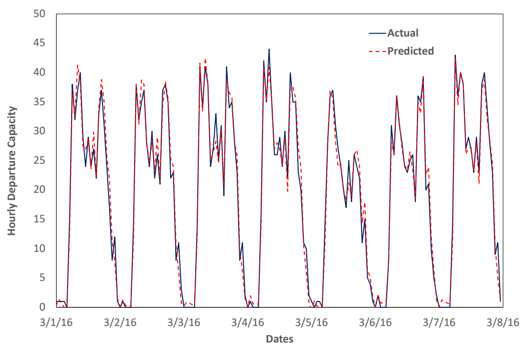
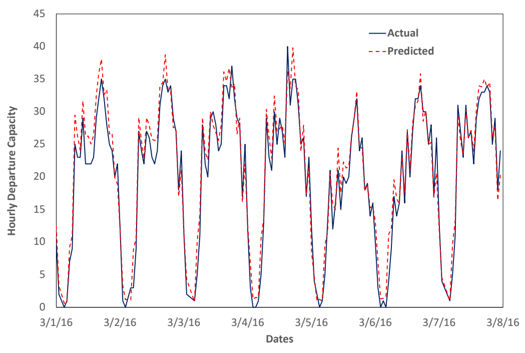


Figure 2.37: Schematic Diagram of Transfer Learning

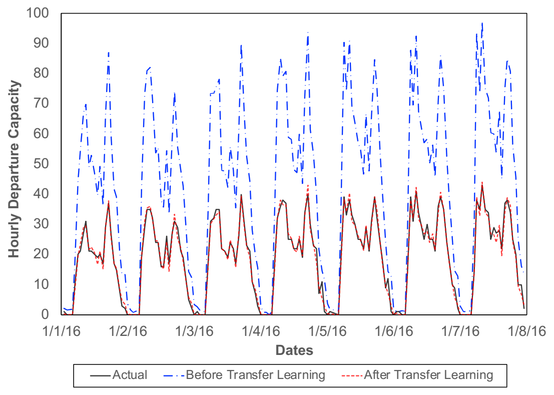


(a) Departure

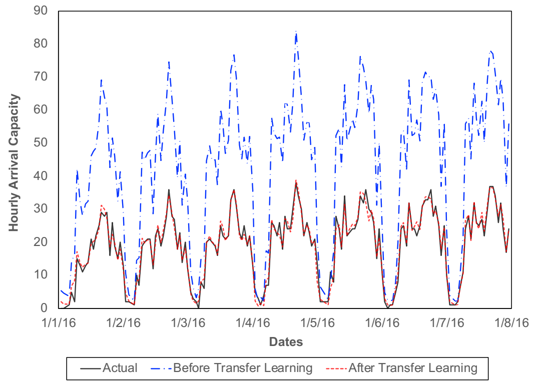


(b) Arrival

Figure 2.38: Prediction of MLP Trained with ATL data for BOS Capacity in March 1^{st} to 7^{th} 2016

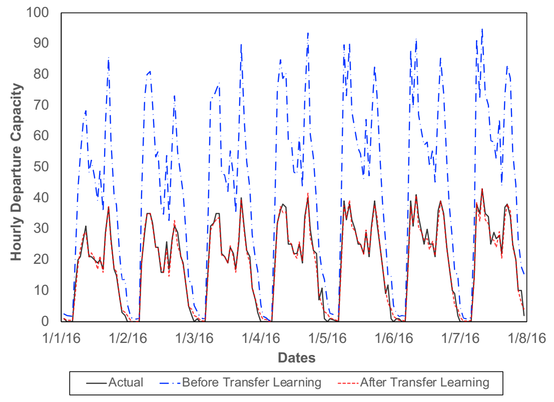


(a) Departure

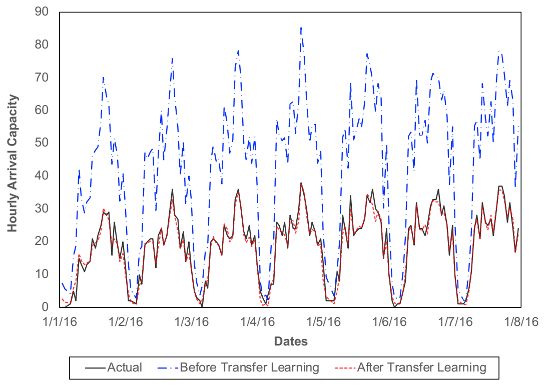


(b) Arrival

Figure 2.39: Actual and Estimated Capacity before and after Transfer Learning of RNN for BOS from January 1st to 7th, 2016



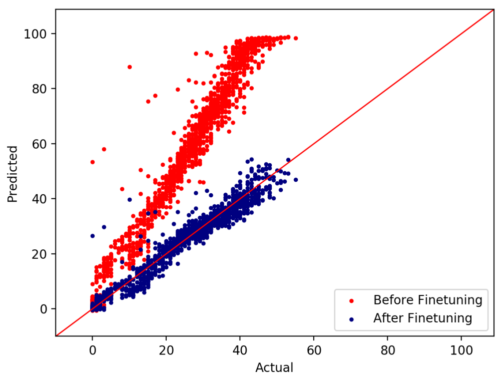
(a) Departure



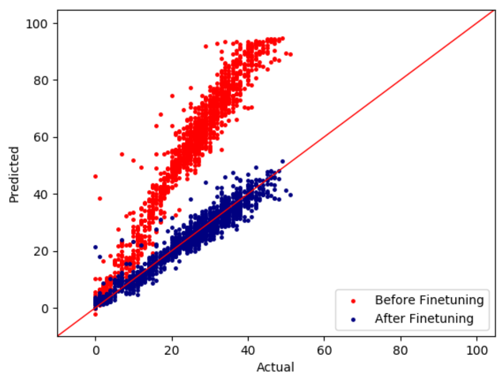
(b) Arrival

Figure 2.40: Actual and Estimated Capacity before and after Transfer Learning of LSTM for BOS from January 1st to 7th, 2016

son is that the BOS capacity prediction performance of the MLP model trained with the ATL data is excellent as shown in Figure 2.38, so that the transfer learning was not necessary. Figure 2.38 and Figure 2.40 plot the actual capacity and predicted values from January 1st to 7th, 2016 to represent the performance of the model for BOS. MLP was reasonably able to identify relevant information in the unseen data without transfer learning, despite different meteorological and geographical conditions.

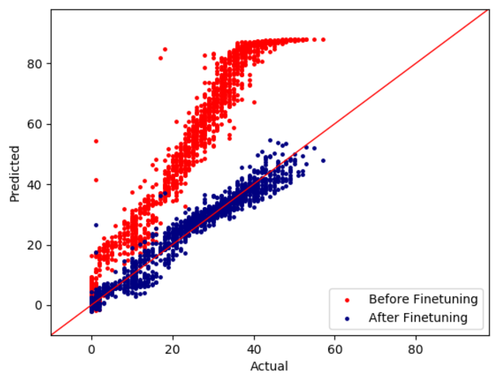


(a) Departure

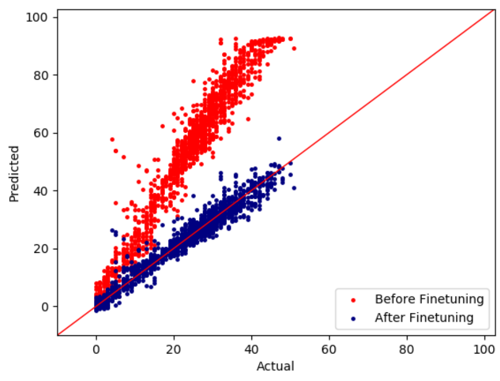


(b) Arrival

Figure 2.41: Predicted vs. Actual of RNN before and after 1 epochs finetuning



(a) Departure



(b) Arrival

Figure 2.42: Predicted vs. Actual of LSTM before and after 1 epochs finetuning

However, RNN and LSTM before transfer learning predicted the departure and the arrival capacity at almost twice the actual value. This difference is due to the training data. The models are trained with the data from ATL, one of the busiest airports in the world. The average hourly arrival capacity of ATL is 50.4, while that of BOS is 22.4. RNN and LSTM seem to be tailored specifically to the ATL data. Nevertheless, transfer learning could turn them into models applicable to BOS. After ten epochs learning with the BOS data, weights of RNN and LSTM were successfully tuned according to the BOS data. In Figure 2.39 and Figure 2.40, we can see that RNN and LSTM can predict the actual value fairly accurately after transfer learning. Furthermore, Figure 2.41, Figure 2.42, Table 2.5 and Table 2.6 show that the knowledge acquired by the pre-trained ATL model can be transferred to the new BOS model with short iteration.

Table 2.5: RMSEs of Hourly Departure Capacity after Finetuning for BOS

Model	Before Finetuning	After Finetuning		
		1 Epoch	5 Epochs	10 Epochs
RNN	39.22	3.07	2.73	2.62
LSTM	37.65	3.32	2.65	2.35

Table 2.6: RMSEs of Hourly Arrival Capacity after Finetuning for BOS

Model	Before Finetuning	After Finetuning		
		1 Epoch	5 Epochs	10 Epochs
RNN	45.37	2.96	2.52	2.35
LSTM	43.08	2.91	2.54	2.39

The effect of transfer learning on the performance of RNN and LSTM can be analyzed in relation to Figure 2.43 and Figure 2.44. For the departure and the arrival capacity of BOS, Figure 2.43 and Figure 2.44 compare the RMSEs of the fine-tuned models in dotted lines with the models built from scratch in solid lines. From the comparison, the fine-tuned models outperformed the models trained from scratch and converged quickly to lower RMSE values. The RMSE of models trained from scratch began at a large value and went down through training, whereas the RMSE of transfer learning models was very low from the first iteration. Therefore, it can be inferred that transfer learning can provide a better starting point for model parameters than random initialization. Based on this initialization, it was possible to improve the accuracy of the model for the target task by fine-tuning the parameters of the pre-trained network to match the new dataset.

Even during the subsequent iteration, the RMSE curve of the models from scratch could not surpass the RMSE of the fine-tuned models. When training models for the same number of iterations, a model that reaches the lower error first is more efficient. In this regard, the use of fine-tuning is advantageous regarding efficiency as well as performance. The knowledge gained by the pre-trained ATL model was successfully transferred to the new BOS model. The pre-trained networks on original data capture common trends that are useful in the target airports. Then continuing back-propagation with the target data teaches the pre-trained model specific features for the target airport.

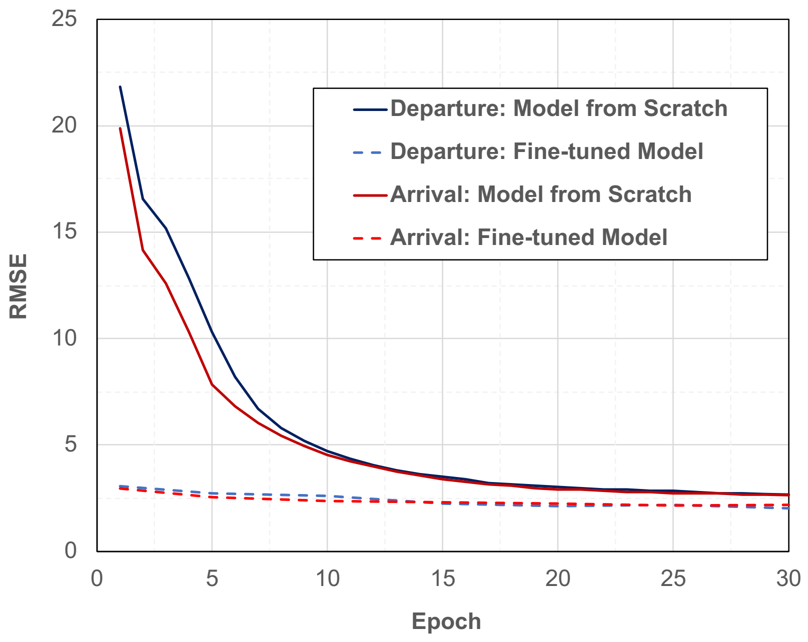


Figure 2.43: RMSE of fine-tuned RNN and RNN from scratch for BOS

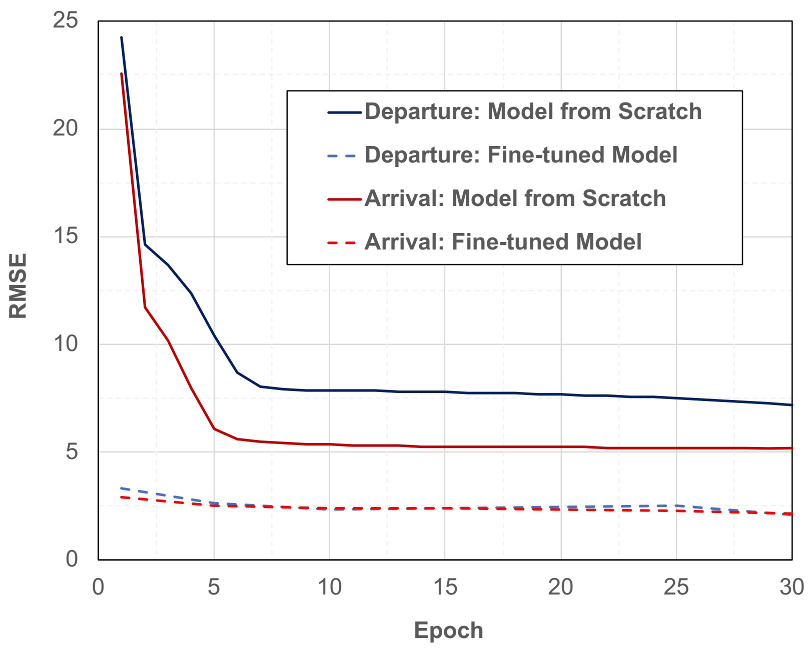


Figure 2.44: RMSE of fine-tuned LSTM and LSTM from scratch for BOS

2.10 Chapter Summary

In this chapter, the airport capacity prediction problem is discussed, and a new approach is proposed through the analysis of existing methods. The artificial neural networks proposed in this study are multi-layer perceptron (MLP), recurrent neural network (RNN), and long short-term memory (LSTM). Hypothesis 1 stated that the proposed models could improve accuracy and computational efficiency compared to the conventional methods for predicting hourly airport capacity. To test hypothesis 1, the models were trained with weather data and traffic data of Hartsfield-Jackson Atlanta airport (ATL) from 2013 and 2017. Also, the models were validated using ATL data for 2018, which was not used in training. The performance of the trained models was evaluated by various factors such as RMSE, actual vs. predicted plot, and cumulative distribution function of mean relative error. In the performance evaluation, the models presented decent predictive performance and successfully predicted the test data as well as the training data. Furthermore, the generalizability of the models for other airports was assessed by using the Boston Logan International Airport (BOS) data. MLP trained with ATL data was able to predict the capacity of BOS without any post-processing. On the other hand, RNN and LSTM trained with ATL performed well after several fine-tuning iterations. There were several advantages of transferring knowledge from the ATL model to the BOS model. The fine-tuned BOS models could quickly converge than the models trained from scratch and were excellent in accuracy.

CHAPTER 3
INDIVIDUAL DELAYS PREDICTION

One of the main concerns of stakeholders in air traffic is punctuality because it is directly related to low productivity [23]. When a plane landed before the ground staff and infrastructure were ready, passengers and crew would be trapped in the cabin. Conversely, if the plane lands later than expected, the operational cost will rise because the ground staff and equipment for landing must wait [124].

3.1 Backgrounds

Especially a flight delay is an undesirable output from the air traffic system but a good indicator of air traffic system efficiency at the same time [1]. The on-time performance of flights has been an important research subject as demands for air travel increase. Thus, there have been attempts to analyze characteristics of airline delays and discover patterns in air traffic. This chapter will examine how existing studies analyzed flight delays.

3.1.1 WITI

Klein et al. have focused more on weather and presented a delays prediction model established on the metric called Weather Impacted Traffic Index (WITI). It is a new predictive linear regression model for estimating airport delays using weather forecast data. As shown in the following equation, the delay minutes from $WITI$ is the weighted sum of $EWITI$, $TWITI$, and $QDelay$.

$$Delay\ Minutes = EWITI \times w_{EWITI} + TWITI \times w_{TWITI} + QDelay \times w_{QDelay}$$

In the expression, $EWITI$ is the en-route convection weather component, $TWITI$ is the terminal convection weather component, and $QDelay$ is the queueing delay component. WITI measures the severity and the impact of weather on the air traffic system [125]. Another characteristic of WITI is that it determines the expected impact of weather forecasts on the scheduled air traffic [126]. Because it is a simple regression, it is easy for users to understand how the model got these results. In this study, 80-85% accuracy was achieved.

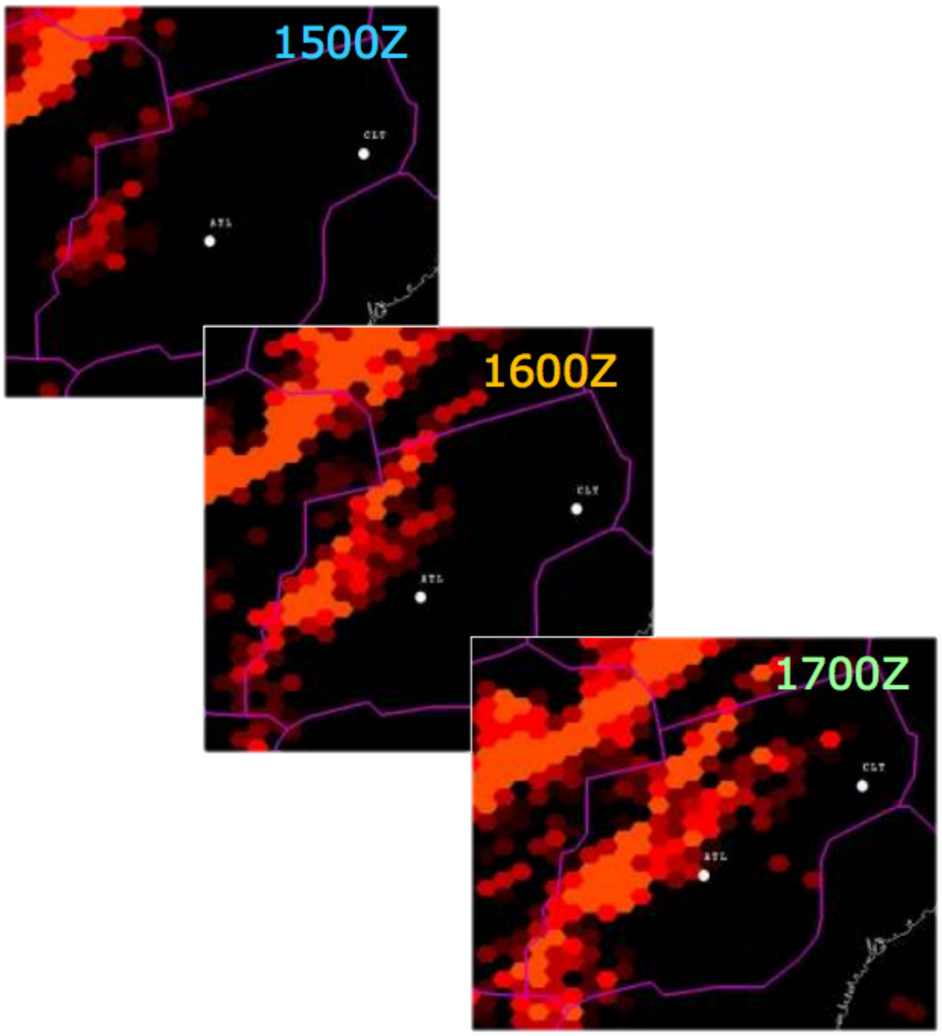


Figure 3.1: Convective Weather Situation from WITI [127]

3.1.2 Statistical Model

The statistical method encompasses the use of regression models and correlation analysis [1]. Abdel-Aty et al. identified the periodic patterns of arrival delays using mathematical frequency analysis techniques [128]. The main contribution of this work is that the proposed frequency analysis method can be used to effectively detect multiple periodic patterns of arrival delays by eliminating noise that may exist in the observed delays data.

Additionally, Pathomsiri et al. used a non-parametric function to assess the efficiency of US airports in terms of delays [129]. Furthermore, Tu et al. estimated departure delays distributions and used the distribution in a strategic departure delays prediction model [130]. However, because the air traffic system has aleatory uncertainty [131], it is not easy for analytical methods to reduce the gap between the actual and the prediction.

3.1.3 Simulation-based Model

For air traffic network, Kim et al. developed a simulator that relies on parallel simulation techniques and synchronization algorithm [132]. The major objects of the air traffic network (ATC, TRACON, ARTCC and flight) are modeled as individual agents that mimic the patterns of activity in real air traffic systems. By simulation-based model, various scenarios are explored under external conditions [132]. Additionally, Yuan developed a new approach to enhance the dispatch reliability of Australian X Airline's fleet. Yuan employed the computer-aided numerical simulation of departure delays distribution and related cost. As a result, the flight schedule optimization is achieved [45].

3.1.4 Deep Learning Model

More recently, a deep learning algorithm is implemented to predict flight delays by Kim [76]. His model is two-fold. The first stage predicts the daily delays status using LSTM, and the result is fed into the second stage. The second stage predicts the delays of a single flight. The models accuracy with the number of layers is in the figure on the bottom. The deep learning architecture improved the prediction accuracy up to 87%. The disadvantage of the deep learning model is that it requires intensive computation. Also, this method does not provide insight to help you understand the results of the model.

3.2 Problem Definition

The objective is to predict arrival delays for individual flights in the United States. The delays here are defined as the difference between the actual arrival time and the scheduled arrival time when the actual arrival is at least 15 minutes later than the scheduled arrival time. To better define the problem to solve, we will inspect the nature of the flight delays prediction problem in detail. The individual delays prediction problem has the following characteristics:

- Imbalanced data: the number of on-time flights is three to four times more than that of delayed flights. If a model is trained with imbalanced data, the model can be biased toward a majority class. Thus, special care must be taken to avoid bias in the model.

- Unknown governing equation: The occurrence of delays is inevitable results from a series of complex event. The various factors like terminal weather, en-route weather, traffic congestion, runway layouts are complex entangled. For these reasons, airline delays prediction is very difficult to solve analytically because the exact form of the function, $occurrence\ of\ delays$ is not known.

$$occurrence\ of\ delays = f(weather, congestion, runway\ layout, etc.)$$

- : A vast amount of data: One of the important points in delays prediction is how to handle a vast amount of data. In an effort to better understand the entire flight ecosystems, huge volumes of data from commercial aviation are collected every moment and stored in databases. The famous air traffic database is Transtat of the Bureau of Transportation Statistics (BTS). For weather, there is the National Oceanic and Atmospheric Administration (NOAA)'s Integrated Surface Database.

By focusing on those facts, the second research question is introduced.

Research Question 2

What is an appropriate method to accurately predict the on-time performance of aircraft?

As an equation, learning a classifier for individual delays prediction is to find a conditional probability distribution, $p(Y = y|X)$, where X is a feature vector for each flight and Y is a set of class label. Each X_i is the normalized number in preprocessing steps, and y_i is a binary variable that indicates the occurrence of delays. For a sample X, the classifier looks for the probability that the sample belongs to the $on\text{–}time$ or $delays$ class. The probability that a sample belongs to $on\text{–}time$ plus the probability of belonging to $delays$ is 1. When $p(Y = 1|X)$ is greater than the threshold value, the sample X is classified as $delays$:

$$\delta(X) = I(p(Y = 1|X) > \tau)$$

The models created through this study is parameterized by Θ.

3.2.1 Possible Improvements with Machine Learning

The gap to be filled in existing methods is identified as follows:

- Inability to model nonlinearity
- Inability to predict individual delays
- Low interpretability
- Intensive computational cost

Though artificial neural networks are very popular and it is used in various fields, there are limitations that make artificial neural networks impossible to apply to all areas. Due to the relative opacity of artificial neural networks, it is often difficult to interpret and is often referred to as a black box. Artificial neural networks have millions or billions of parameters in the learning system, and it is not easy to identify their meaning [133]. Artificial neural networks algorithms also depend on a large number of labelled examples and are most effective when there are thousands, millions, or billions of training samples [134]. On the other hand, since the regression method is a linear model, only linear relationships between variables can be found. Moreover, it aggregates delays minutes at airport level, so it is not able to predict individual delays.

The algorithms that can perform the task of classification are shown in Table 3.1. Considering the comparison of Table 3.1, we first considered ensemble methods which appeared to be optimal. The next best, k-Nearest-Neighbors, was considered further. There are several reasons to explain why machine learning was tried.

A number of previous studies have demonstrated the applicability of the machine learning methods. Rebollo and Balakrishnan have created variables indicative of the state of the NAS. They predicted network-related delays of the future by utilizing the system-level dependencies among airports [135]. Zonglei et al. have trained classification models to find how severe daily delays are at the hub-airport of China in the future [36]. Not only the unexplicit relationship between weather elements and delays but also uncertainty in weather forecast made the problem harder to be solved. Thus, supervised machine learning algorithms and probabilistic methods are implemented to resolve those concerns. It is worth millions of dollars a year to ensure that stakeholders plan accordingly by providing the correct time of arrival [124].

Other common advantages of machine learning are the followings. First of all, machine learning algorithms have the insight to interpret results from the model, unlike deep learning. In order to convince users of the model results, you should be able to explain how this is determined to users. The reason for the specific decisions is important to users, especially decision makers in data mining domains. Without knowing why this rule was created, the decision maker can not actually use it.

Secondly, the volume of historical flight and weather data are too large to analyze analytically. Moreover, relationships between causal factors and delays or even correlations among factors are extremely complicated and highly nonlinear to test all hypothesis. Machine learning is able to develop models vigorously with a huge amount of dataset, and it has the ability to discover and display the hidden patterns in the data. Machine learning algorithms also take less time to train from seconds to hours. In summary, machine learning is a clever method that can bridge the gap between analytical and deep learning methods.

In light of this characteristic of the individual delays prediction, it seems that the ensemble methods are suitable methods for this problem. By analyzing a large amount of data through the principle of machine learning, it will be possible to search for relationships, patterns, and rules existing in these data, and to extract meaningful knowledge by modeling them. This can be formulated as the second hypothesis:

Hypothesis 2

Flight delays prediction using the ensemble methods of machine learning provides interpretability and imbalance data handling, but accuracy is as good as the existing methods.

To validate Hypothesis 2, experiment 2 is planned to be executed. For a comparison assessment of machine learning algorithms, the accuracy and computational efficiency will be measured. Here are some questions to be answered by experiment 2 in particular:

- Validation of optimal hyperparameters for each algorithm using Grid Search
- Demonstrate which machine learning algorithm is better to classify individual flight's delays
- Comparison of computational time to train models
- How does the performance of the classification model change with threshold values?
- What is the performance of the model different for the validation data when compared to the training data?

A set of validation criteria for experiment 2 is the following:

- Accuracy $\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$
- Confusion Matrix
- Receiver Operating Characteristic (ROC) Curve

Hypothesis 2 will be accepted

- If the accuracy of the machine learning methods is at the level of the existing model.
- If the machine learning methods provide better interpretability of results than existing models.

3.3 Methods

Supervised machine learning classification algorithms are implemented with the help of the $scikit-learn$. The $scikit-learn$ is an open source machine learning library for the Python programming language [136].

3.3.1 Decision Tree (DT)

Decision tree method recursively partitions the input dataset at a node for a chosen attribute. Dataset is partitioned into mutually exclusive subsets in each split. The strategy of selecting the attribute for a split is to find a partition that can optimize the quality of a split. Criteria for determining the quality of the split could be Gini impurity or entropy. A detailed description of Gini impurity and entropy can be found in Appendix D.

A node uses one of its attributes as its decision criterion to partition the data set. Possible values of the chosen feature at a node are shown as branches. The branches are determined to minimize the loss at the node. The loss at the node can be expressed as the sum of T_L and T_R:

$$Loss(T) = T_L + T_R$$

where T_L and T_R are the loss of the left and the right branches, respectively.

The algorithm grows the tree using nodes and branches. A leaf node terminates a sequence of nodes and branches. At each leaf node, a local model storing the distribution over class labels is defined. The class of a sample is determined by tracing nodes and branches to the leaf nodes [137]. In the end, the model for the input variable X can be written in the following form,

$$f(\mathbf{X}) = \mathbb{E}[y|\mathbf{X}] = \sum_{m=1}^{M} w_m \phi(\mathbf{X}; \mathbf{v}_m)$$

where M is the number of class labels and w_m is the probability distribution over class labels in the m^{th} leaf. $\mathbf{v}_m$ encodes the choice of variable to split and the threshold value on the path from the root to the m^{th} leaf [138].

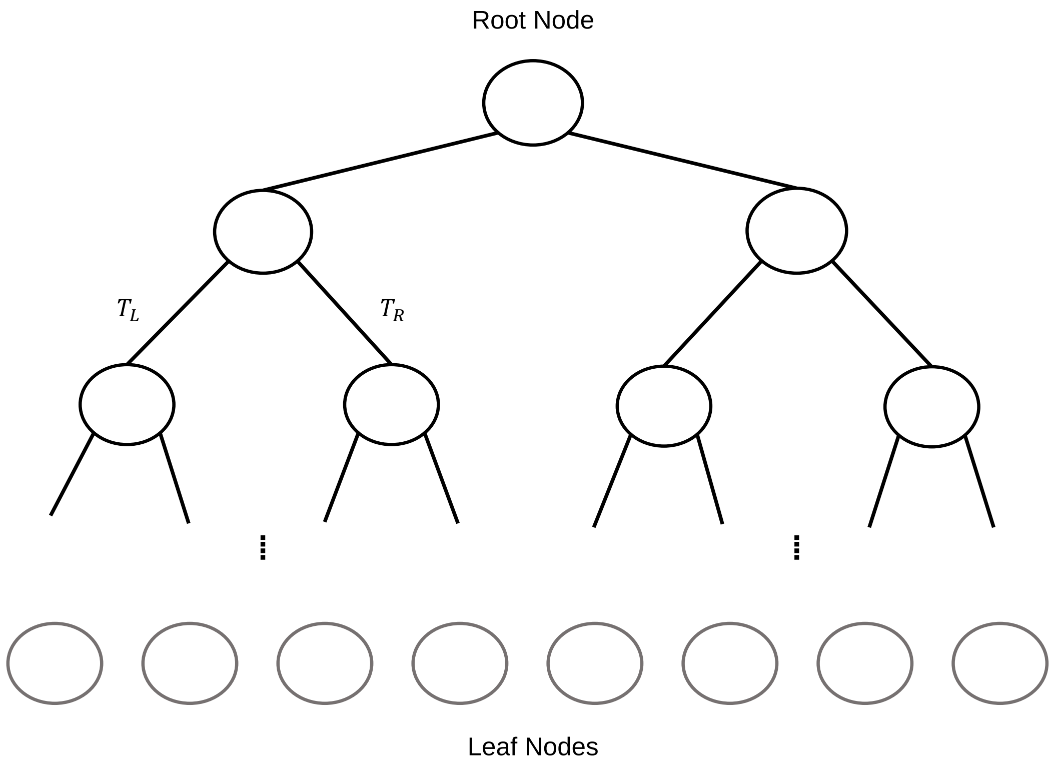


Figure 3.2: Decision Tree

3.3.2 Random Forest (RF)

Random forest creates an ensemble model based on a decision tree [139]. It is bagging of random decision trees. It builds a large collection of de-correlated trees which are noisy but unbiased and averages them to reduce the variance. Then it takes a class vote results from many individual trees and synthesizes them by averaging to reduce the variance. In the last step, it classifies a sample using a majority vote [140]. Let $\hat{C}_b(x)$ be the class prediction of the b^{th} tree, then the class obtained from random forest, $\hat{C}_{rf}(x)$, is

$$\hat{C}_{rf}(x) = \text{majority vote}\{\hat{C}_b(x)\}_1^B$$

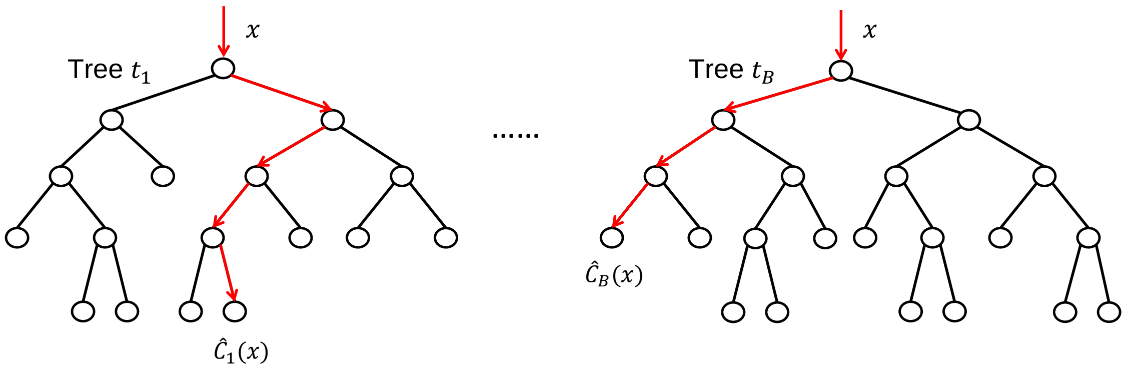


Figure 3.3: Random Forest

3.3.3 AdaBoost

AdaBoost is a shortened form of the adaptive boosting algorithm. It is a method of converting weak classifier into a highly accurate prediction rule. It can improve the performance of a classifier by learning weak classifiers on the weighted example set repeatedly. Its name 'adaptive' means that it focuses on the samples incorrectly predicted by the previous classifier and weighs them heavily when finding a new weak classifier [141]. Thus, the produced sequence of classifiers is dependent on the previous one and focuses on the previous one's errors. After a few sequences of learning, all the weak classifiers are combined linearly and converted into the final classifier with highly accurate prediction rule. AdaBoost is simple and easy to implement as there is no need for prior knowledge about the weak learner.

The error of the i^{th} classifier for the total N samples is:

$$\varepsilon_i = \frac{\sum_{j=1}^{N} w_j \delta(C_i(x_j) \neq y_j)}{\sum_{j=1}^{N} w_j}$$

where C_i is the i^{th} classifier, w_j is the weight of each sample, and y_j is a class label. The weight of each sample is updated as follows:

$$w_j^{i+1} = \frac{w_j^i \exp(-\alpha_i y_j C_i(x_j))}{Z^{(i)}}.$$

The weight of the i^{th} classifier, α_i, is determined from the error of the classifier:

$$\alpha_i = \frac{1}{2} \ln \left(\frac{1 - \varepsilon_i}{\varepsilon_i} \right).$$

Then the final classifier boosting k weak classifiers is

$$C^*(X) = \underset{Y}{\operatorname{argmax}} \sum_{i=1}^{k} \alpha_i \delta(C_i(X) = Y)$$

3.3.4 k-Nearest-Neighbors Classifier (kNN)

To classify a sample X, the k-Nearest-Neighbors classifier (kNN) draws a sphere centered on X enclosing exactly k training samples. It calculates the distance from the point X to its neighboring points and finds k nearest samples. Afterwards, it checks the label of k nearest training points and assigns the label with the majority vote to X [141]. Although it is very simple and straightforward, the k-Nearest-Neighbors classifier succeeded even when the decision boundary is irregular.

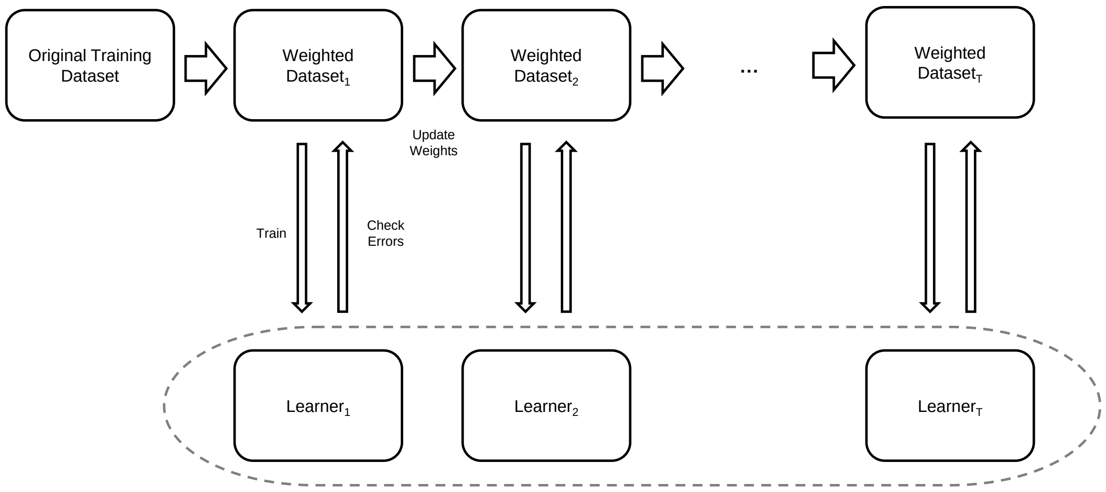


Figure 3.4: AdaBoost

3.3.5 Feature Selection

Feature selection is a process of dimensionality reduction on the dataset for use in model construction [142]. Because the categorical variables have been converted to a vector, the original dataset has more than 50 variables. It will lead to the curse of dimensionality in classification and is hard to facilitate data understanding. The null hypothesis significance testing is conducted to improve the prediction performance. The p-value is the probability that the null hypothesis is true and quantifies the significance of the evidence. For each variable, the p-values are yielded, and their statistical significance was measured. As a result, the top 20 variables that can better explain delays are selected. The p-value for the top 20 variables are in Appendix E.

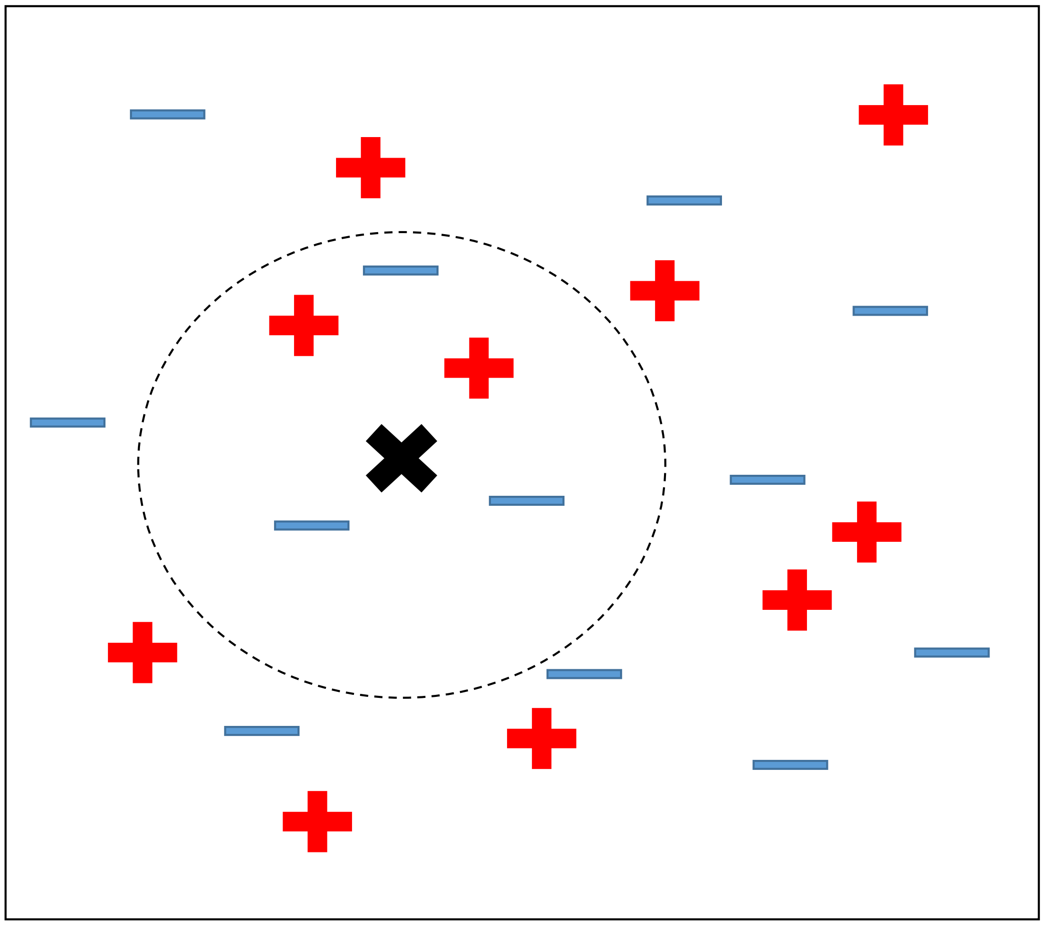


Figure 3.5: k-Nearest-Neighbors

3.3.6 Imbalanced Data Handling

The dataset is described as imbalanced when the classification categories are not approximately equally represented [143]. According to this criterion, the training data is imbalanced because the number of on-time flights is three to four times more than that of scheduled flights. It means that even if you classify all flights as the on-time class, you can have a forecast accuracy of 75% or more. It should be noted that accuracy does not capture the practicality of the model. Another problem that can arise from an imbalance in training data is that the on-time class is overfitted and the delayed class can be ignored. In the real world, identifying the minority class is required as it is costly to misclassify examples from the minority class [144].

There are data-level and algorithm-level methods to resolve problems arising from imbalanced data. Data-level methods adjust data, and algorithm-level methods use algorithms to address issues. Since algorithm-level methods are computationally expensive, we use the data-level approach here. Several data-level approaches are compared in Table 3.2. Based on the comparison of Table 3.2, a mixed form of SMOTE and under-sampling is used to address issues emerged from imbalanced data.

Synthetic Minority Over-sampling TEchnique (SMOTE) is an over-sampling approach that creates synthetic minority class examples. The minority class is over-sampled by introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors [143]. The pseudo-code of SMOTE is described in detail in Algorithm 2 and the schematics of SMOTE example with $k = 5$ and 300% sampling needed is in Figure 3.6. To apply SMOTE for the sample marked with a star in Figure 3.6, we first need to find the five nearest neighbors. Because 300% sampling is needed, we choose three neighbors randomly among the five nearest neighbors. Then virtual lines are drawn between the sample marked with a star and the three randomly drawn neighbors. Three synthetic samples are created by selecting arbitrary positions on each line segments. These processes are repeated on all minority class samples to generate 300% new synthetic samples.

Algorithm 2 SMOTE (T, N, k) [143]

```
Require: Original minority class samples $Sample[\cdot][\cdot]$,
         Number of minority class samples $T$,
         Amount of sampling needed $N\%$,
         Number of nearest neighbors $k$

  $N = int\left(\frac{N}{100}\right)$
  $num\_att$: Number of attributes
  $new\_idx = 0$: Count number of synthetic samples generated
  $Synthetic[\,][\,]$: Array for synthetic minority samples
  for i=1 to T do
    $neighbors\_idx$ : The $k$ nearest neighbors'indices of $i$
    $Populate(N, i, neighbors\_idx)$
  end for
  return $\frac{N}{100} \cdot T$ Synthetic minority class samples

  Function $Populate(N, i, neighbors\_idx)$
  while $N \neq 0$ do
    Choose one of the $k$ nearest neighbors of $i$ by a random integer $nn \in \{1, k\}$.
    for attr=1 to $num\_att$ do
      $diff = Sample[neighbors\_idx[nn]][attr] - Sample[i][attr]$
      $gap$ = random float $\in \{0, 1\}$
      $Synthetic[new\_idx][attr] = Sample[i][attr] + gap \cdot diff$
    end for
    $new\_idx = new\_idx + 1$
    $N = N - 1$
  end while
  return $Synthetic$
```

Table 3.1: Comparison of Classification Algorithms

Criterion	Logistic Regression	Support Vector Machine	Naive Bayes	Ensemble Methods	Artificial Neural Networks	kNN
Computational complexity	×	×	○	○	×	○
Nonlinear decision boundary	×	×	×	○	○	○
Resistant to overfitting?	×	×	○	○	△	△
Interpretability	○	○	○	○	×	○

Table 3.2: Comparison of Imbalanced Data Handling Methods [143]

Criteria	Under-sampling	Over-sampling	Under-sampling & SMOTE
Better generalizable decision surface	×	△	○
Minority class recognition	×	△	○
Computational cost	○	○	○

SMOTE is a powerful solution for an imbalanced dataset that can address drawbacks of the over-sampling and under-sampling. It has been shown that SMOTE improves the performance of classifiers in various application domains [145] [146]. The combination of SMOTE and random under-sampling of the majority class was used in the training step as it resulted in more precise estimation than other sampling methods [143]. Under-sampling randomly removes the majority class samples with different class labels from the majority of k nearest neighbors. SMOTE and under-sampling work in the direction of correcting datasets and learners that are biased toward the majority class. Also, they make the ratio of the number of samples in the minority class and the majority class to a certain value.

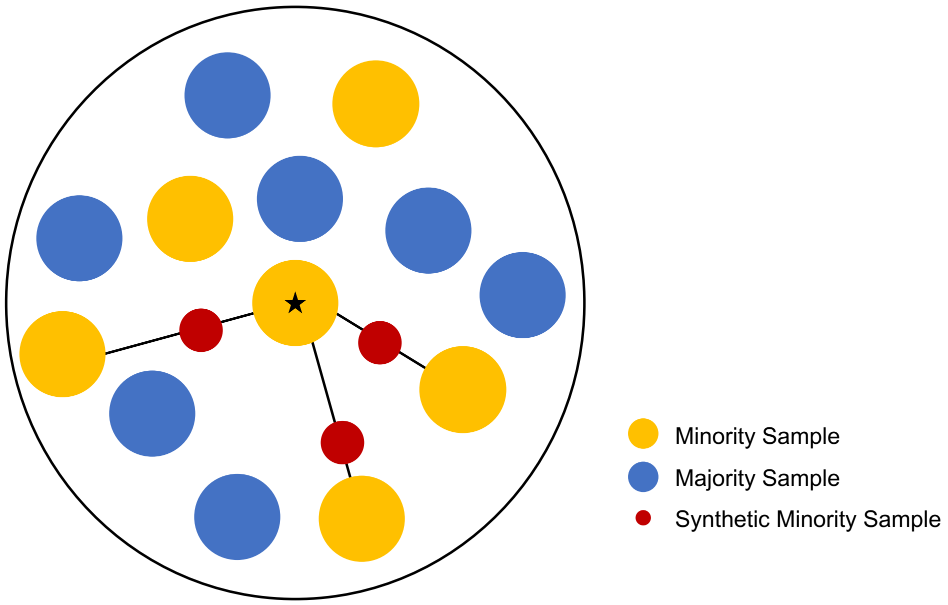


Figure 3.6: Schematics of SMOTE example, 300% sampling needed and $k = 5$

3.4 Data Source and Processing

Data is meaningful when it is handled in an appropriate way and becomes information for stakeholders. This section describes the data collected for the individual delays model and method to handle them. A vast amount of air traffic and weather data is being collected from a variety of sources.

3.4.1 Data Collection

For 45 major airports, US domestic airline traffic data and weather data from 2005 to 2015 are obtained from the Bureau of Transportation Statistics (BTS)' Airline On-time Performance dataset and National Oceanic and Atmospheric Administration (NOAA)'s Integrated Surface Database, respectively.

The historical flight dataset contains on-time arrival data for non-stop domestic flights by major air carriers [147]. It also provides additional information such as origin/destination airports, flight numbers, flight schedules and delay minutes. From the BTS'Airline On-time Performance dataset, the following data fields are extracted for each individual flight because those are factors having impacts on flight delays [128].

- Origin
- Destination
- Quarter of Year
- Month
- Day of Month
- Day of Week
- Scheduled Departure Time in Local Time

- Scheduled Arrival Time in Local Time
- Arrival Delays Indicator

Arrival delays indicator is used as a label of an example. It is 0 if actual arrival time minus scheduled arrival time is less than 15 minutes or 1 if actual arrival time minus scheduled arrival time is greater than or equal to 15 minutes.

On the other hand, NOAA's database contains historical hourly surface-based weather observations including wind, cloud height, visibility, temperature, pressure, and precipitation reported approximately hourly at worldwide stations [148]. The following items are obtained from the historical weather dataset.

- Wind Direction Angle [deg]
- Wind Speed Rate [m/s]: A strong sidewind or tailwind increases the risk of landing and landing distances. They are factors that causes delays because they sometimes make runways unavailable [131].
- Visibility [m]: When fog, haze, smoke, and heavy precipitation reduce the visibility, an aircraft must be at a sufficient distance from other aircrafts and operate limitedly [131].
- Precipitation [mm]
- Snow Depth [cm]
- Snow Accumulation [cm]
- Weather Intensity Code
  1:Light, 2:Moderate, 3:Heavy, 4:Vicinity

- Weather Descriptor Code

  1:Shallow, 2:Partial, 3:Patches, 4:Low Drifting, 5:Blowing, 6:Showers, 7:Thunderstorms, 8:Freezing

- Precipitation Code

  1:Drizzle, 2:Rain, 3:Snow, 4:Snow Grains, 5:Ice Crystals, 6:Ice Pellets, 7:Hail, 8:Small Hail and/or Snow Pellets, 9:Unknown Precipitation

- Obscuration Code

  1:Mist, 2:Fog, 3:Smoke, 4:Volcanic Ash, 5:Widespread Dust, 6:Sand, 7:Haze, 8:Spray

- Other Weather Code

  1:Well-Developed Dust/Sand Whirls, 2:Squalls, 3:Funnel Cloud, Tornado, Waterspout, 4:Sandstorm, 5:Duststorm

- Combination Indicator Code

  1:Not part of combined weather elements, 2:Beginning elements of combined weather elements, 3:Combined with previous weather element to form a single weather report

The above items are based on the fact that flight delays often occur in association with convective weather in the terminal area. Also, low ceiling/visibility conditions, high surface winds and precipitations make an aircraft landing difficult [149] [128].

For the test data, 2016's Flight schedule and weather forecast are collected via API and merged to create a test set for the prediction process. Current flight data not yet released from the BTS' is available in FlightStats APIs [150]. Also, the weather forecast is obtained from World Weather Online API. The forecast for the origin and the destination airport at scheduled departure and arrival time is accessible through their weather APIs [151].

3.4.2 Data Preprocessing

Historical flight data and weather data are joined by using origin airport, destination airport and scheduled time as the joining keys. That way, each row of the joined table consists of flight schedule and corresponding terminal weather of a flight. Then, the merged dataset is subjected to a preprocessing step.

By preprocessing, all of the categorical variables are converted to a one-hot vector since machine learning algorithms exhibit better performance with numerical variables. For instance, $Day\ of\ Week$ is a categorical variable that can have seven kinds of values from Monday through Sunday. Thus, it is a vector with 7 components in the numeric form. When it is Monday, the variable $Day\ of\ Week$ is converted to $[1, 0, 0, 0, 0, 0, 0]$ and Tuesday is $[0, 1, 0, 0, 0, 0, 0]$.

The canceled and diverted flights in the training set are deemed as delayed. To deal with missing values in weather data, linear interpolation is used with two adjacent known values. Furthermore, features are normalized and the range of features are scaled to prevent the model from being dominated by a few features that have a high variance.

3.5 Model Evaluation Criteria

3.5.1 10-fold Cross-Validation

Cross-validation is a method that can estimate a model's accuracy on unseen data. A dataset is divided into 10 approximately equal-sized subsets. The k^{th} $(k = 1, ..., 10)$ subset is chosen to be a validation set and set aside for testing. The rest subsets are utilized as a training set and they build a model. At last, the cross-validation error (CV) can be calculated from the following equation:

$$CV = \frac{1}{10}\sum_{i=1}^{10} L(y_i, \hat{f}^{-\kappa(i)}(x_i))$$

where (x_i, y_i) is the i^th subset. $\hat{f}^{-\kappa(i)}(x_i)$ is a fitted function with samples $(x_j, y_j), j \neq i$ [152].

3.5.2 Performance Measure of Classifiers

As described in Section 3.3.6, it is not sufficient to measure the performance of a classifier with accuracy when it is an imbalanced dataset. This is because the classifier can have decent accuracy even when a classifier labels every sample as the on-time. Thus, other metrics should be considered. Typical metrics used in the machine learning community are Recall, Precision, and F1 scores. The following terms are defined with the same convention as it used in the confusion matrix of Table 1.2:

- True Positive (TP): number of positive samples that are labeled as positive
- False Positive (FP): number of negative samples that are labeled as positive
- True Negative (TN): number of negative samples that are labeled as negative
- False Negative (FN): number of positive samples that are labeled as negative

Using the terms outlined above, Recall, Precision and F1 scores can be described as follows [153]:

- Recall: The proportion of true positive (TP) to the number of positive samples

  $Recall = Sensitivity = TPR = \dfrac{TP}{(TP + FN)}$

- Precision: The proportion of true positive (TP) to the number of samples that are classified as positive by the classifier

  $Precision = Confidence = \dfrac{TP}{TP + FP}$

- F1 score: The weighted average of Recall and Precision

  $F1\ score = \dfrac{2TP}{(2TP + FP + FN)}$

A common disadvantage of Recall, Precision and F1 score is that the ability to recognize the negative class is ignored. The ROC curve is proposed to overcome this weakness and measure the general performance. By using the ROC curve, we will be able to evaluate the performance of a classifier considering the ability to detect the negative class as well as the positive class.

3.5.3 Receiver Operating Characteristic (ROC) Curve

The Receiver Operating Characteristic (ROC) curve is the plot for a binary classifier's performance. As shown in Figure 3.7, It illustrates True Positive Rate (TPR) vs. False Positive Rate (FPR) for a set of a threshold, τ. Let $\delta(x) = I(f(x) > \tau)$ be the decision rule, where $f(x)$ is a measure of confidence that $y = 1$. A threshold of $+\inf$ matches up with (0,0) of ROC curve, and $-\inf$ produces (1,1). The lower the threshold, the higher the TPR, while at the higher the threshold, FPR is expected to be lower. This is because when the threshold is low, the positive class can be predicted more easily.

Here, TPR, also known as the sensitivity, and FPR are computed as follows [138]:

$$TPR = \frac{TP}{(TP + FN)} \approx p(\hat{y} = 1 | y = 1)$$

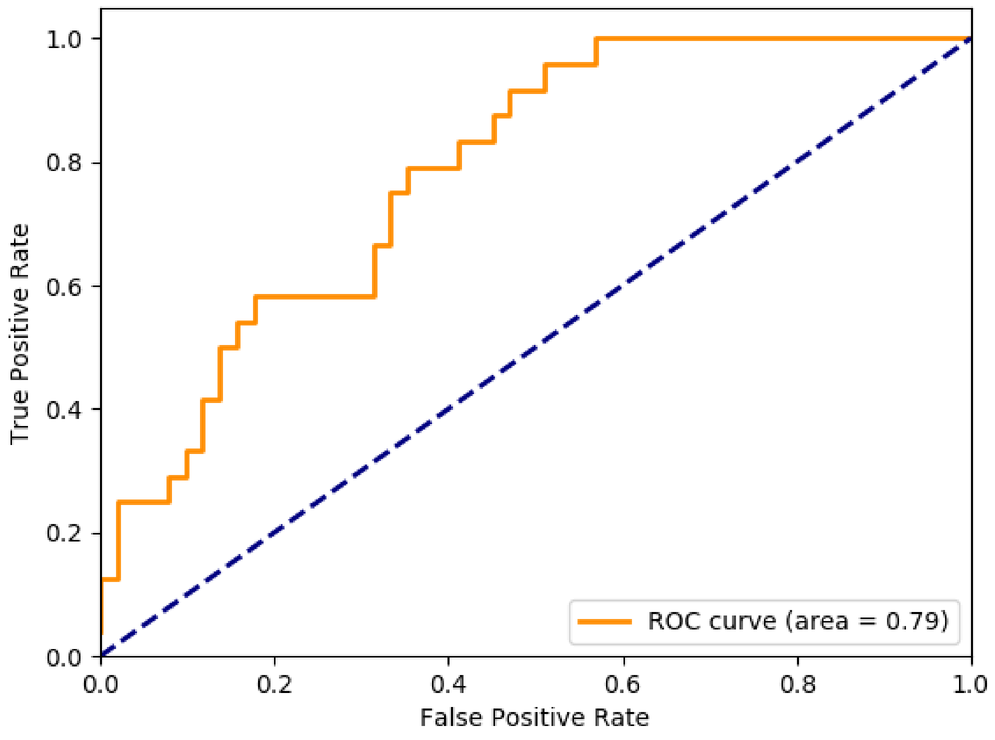


Figure 3.7: Receiver Operating Characteristics Example [154]

$$FPR = 1 - \text{sensitivity} = \frac{FP}{(TN + FP)} \approx p(\hat{y} = 1|y = 0).$$

The ideal point on the ROC curve is (0,1). The point is where all positive samples are correctly classified, and no negative examples are misclassified as positive [143]. Therefore, the closer the curve is to (0,1), the better the performance. The diagonal line connecting (0,0) and (1,1) means random guessing. If it is close to or below the diagonal line, it is a model with worse performance than the random guess. Plus, the Area Under the ROC Curve (AUC) is another strict measure of the predictive performance when classifying an imbalanced dataset. A larger AUC indicates a better prediction.

3.6 Experiments

In this research, we had focused on arrival delays of individual flights using supervised machine learning algorithms. To predict delays of individual flights, supervised machine learning algorithms were implemented with features including flight schedules and weather conditions at the origin and the destination. In order to increase the predictive capability, models were trained for individual origin-destination (OD) pair not for the entire National Airspace System (NAS) by capturing each airports weather characteristics originated from a geographic location. In the prediction step, flight schedule information combined with weather forecasts was fed into the model to get the predictions on scheduled flights not yet flown.

The overall process of individual flight delays prediction from data collection to class label generation is shown in the flowchart in Figure 3.8. The flight and weather data are collected from the BTS and the NOAA, respectively. Each flight data is associated with weather data at the origin and destination airports. The merged data is preprocessed as described in Section 3.4. Then they are split into training and testing data. The training data goes through the sampling step. SMOTE sampling generates a given number of new instances by sampling and interpolating minority samples. After the classes are balanced, supervised learning models are trained with the training data. The trained model can be used to perform classification tasks on test data.

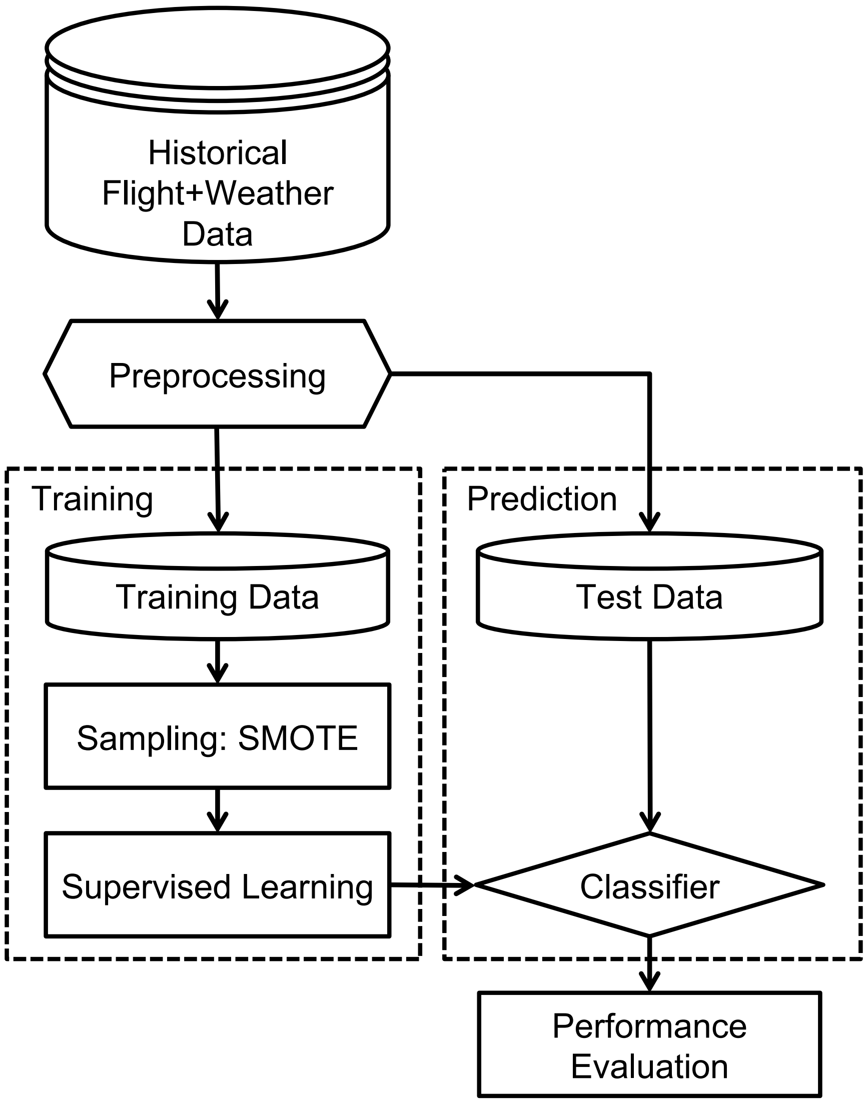


Figure 3.8: Flow Chart for Individual Flight Delays Prediction

3.7 Results

3.7.1 Tuning Models

This section discusses the training results of the classification model by Random Forest, AdaBoost, and k-Nearest-Neighbors. To begin the classification, the models need to be optimized. The models were optimized through designs of experiments (DOE). Since there are different hyperparameters for each algorithm and feasible values for each algorithm, a few of possible hyperparameters for each model were selected and tested.

Table 3.3 lists the results of adjusting the key parameters of Random Forest. The number of trees determines how many decision tree classifiers be fitted. Also, the maximum depth of the tree is about how deeply the trees expand. If the maximum depth is not limited, the algorithm repeats partitioning the node until one sample is left in each leaf. However, given the minimum number of samples per leaf, there must be more than the given number of samples per node.

Max Depth	Num Trees	Min Samples/Leaf	Test Accuracy	Elapsed Time (sec)
Not limited	200	1	0.8502	9

Table 3.3: Test Accuracy and Elapsed Time to Train of Random Forests with Parameter

Table 3.4, on the other hand, shows the result of parameter tuning for AdaBoost. AdaBoost focuses on misclassified samples and iteratively fits the base estimator. Random Forest is considered as a base estimator of AdaBoost. According to the experimental results, the most accurate AdaBoost can be achieved by training 50 Random Forests repeatedly.

Finally, Table 3.5 is the results of parameter tuning for kNN. Experiments were conducted on how many neighbors to consider to classify a sample. The experimental results showed the highest AUC and accuracy when considering the nearest four neighbors. The

Base Estimator	Num Estimator	Test Accuracy	Elapsed Time (sec)
Random Forest	50	0.8444	12

Table 3.4: Test Accuracy and Elapsed Time to Train of Adaboost with Parameter

accuracy of kNN was lower than the other two classifiers, but the training speed was much faster.

10-folds cross-validation was performed to estimate the test accuracy of the trained model. For 10-folds cross-validation, the entire dataset was partitioned into ten small sets, one of which was used as a test set and the remaining nine as a training set. During ten iterations, each set becomes a test set once in turn. Then the ten results are averaged. The evaluation indicators used here are a receiver operating characteristic (ROC) curve and the Area Under the ROC curve (AUC) because the data is imbalanced. Therefore, the performance of the classifier can be evaluated by averaging ROC and AUC obtained from ten experiments.

Figure 3.9, Figure 3.10, and Figure 3.11 are ROC curves and AUC of Random Forests, AdaBoost, and k-Nearest-Neighbors from 10-folds cross-validation, respectively. The blurry lines are each case of 10-folds cross-validation, and their average is the blue line. Also, the shaded area indicates an area deviated from the average by the standard deviation of the curve. The average and standard deviations of the AUC values, as well as the curves, were obtained. Compared with Figure 3.9 and Figure 3.10 comprehensively, Random Forests

Num Neighbors	Test Accuracy	Elapsed Time (sec)
4	0.8112	1

Table 3.5: Test Accuracy and Elapsed Time to Train of k-Nearest-Neighbors with Parameter

and AdaBoost have almost the same performance in terms of AUC and ROC.

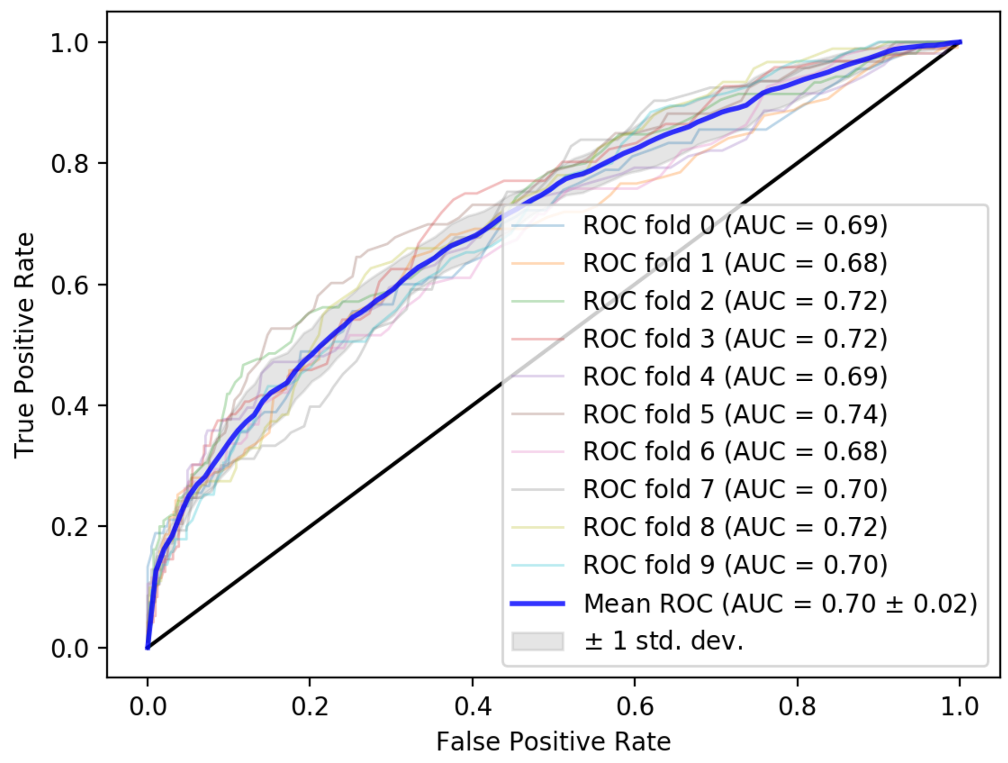


Figure 3.9: ROC from 10-folds Cross Validation of Random Forests

An example of the decision rule for each tree-based classifier can be visualized as shown in Figure 3.12 and Figure 3.13. In each figure, the dots represent the samples to be classified, and the actual class of each sample is presented by the color filled. For simplicity, Random Forest consists of ten Decision Tree estimators and AdaBoost consists of ten Random Forest estimators. The decision surfaces learned by each estimator are divided into blue and red. The original data has much more dimensions, but the model is visualized only with wind direction and wind speed for convenience.

The decision surfaces divide the space into two classes. The final decision surfaces of the two classifiers are obtained by superimposing the results from ten trees and are shown at the end of Figure 3.12 and Figure 3.13. Since Decision Tree is a nonlinear mapping of fea-

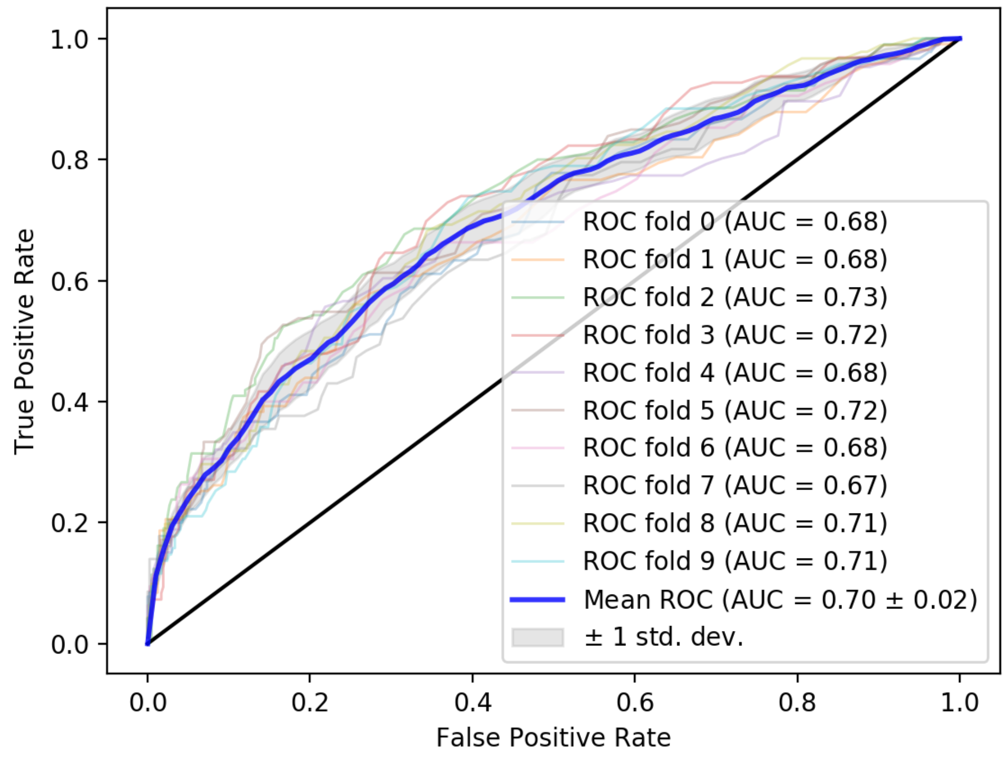


Figure 3.10: ROC from 10-folds Cross Validation of AdaBoost

tures to a class, not only the decision surface from each estimator but also the superimposed final decision surface is also nonlinear.

Figure 3.14 plots one of the Decision Trees that make up the example Random Forest. The sample class is determined by tracing nodes and branches from the top node. The local model of the class distribution stored in the leaf node determines the class of a sample. For example, starting from the root nodes, the samples that satisfy the following conditions are assigned to the leftmost leaf node:

(1) Normalized visibility at the destination airport ≤ 0.101, and

(2) Normalized wind direction at the origin airport ≤ 0.111, and

(3) Normalized visibility at the origin airport ≤ 0.101.

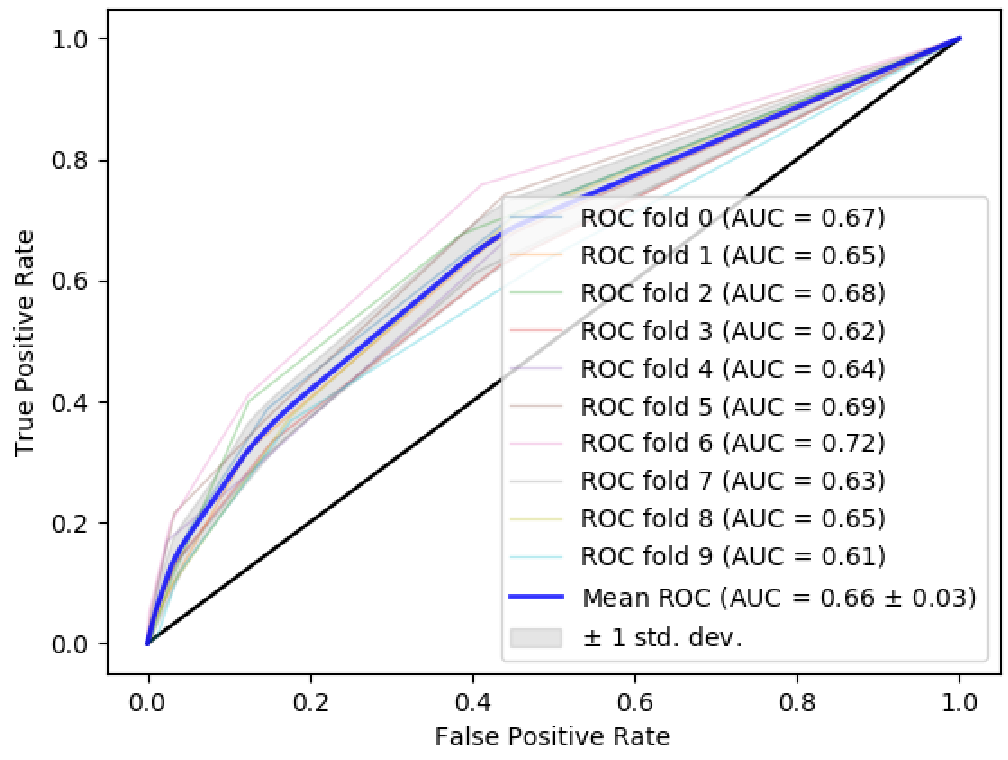


Figure 3.11: ROC from 10-folds Cross Validation of kNN

Samples arriving at the leftmost node in this way are classified as on-time or delayed with a probability of $14 : 142$.

Each tree finds a variable and its value that can be used to create an optimal split based on the criteria in Appendix D. The color of each node indicates which class has more samples at that node. As the color of the node becomes dark blue, the delayed class is more dominant. In contrast, the dark orange means a node with the on-time class dominance. The higher the color saturation, the more the class dominates the other class. It is preferable that the color becomes thicker as it goes to the leaves.

Observing the variables selected from the root node to the leaves, the variables used to determine the optimal partition were visibility, wind, and precipitation at the origin and the destination airport. This result can be interpreted in conjunction with the weather variables in Section 3.7.3. In Figure 3.23 and Figure 3.24 of Section 3.7.3, we can see that the probability of delays increases significantly with the change of the corresponding weather variable. Briefly, these variables are important for predicting the occurrence of delays. In Section 3.7.3, we will discuss the importance of weather variables in more detail.

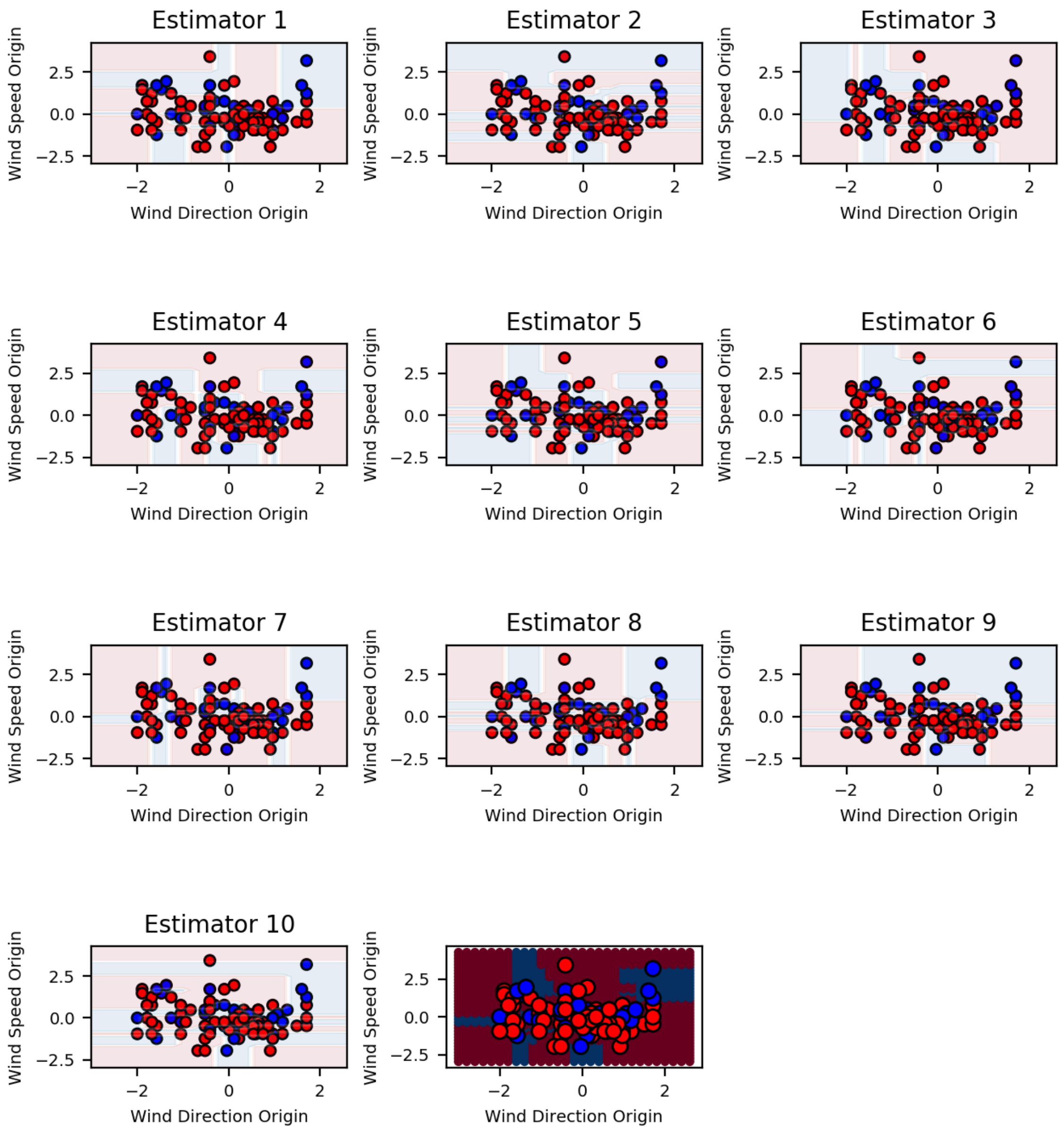


Figure 3.12: 10 Estimators and Final Decision Boundary for Random Forests

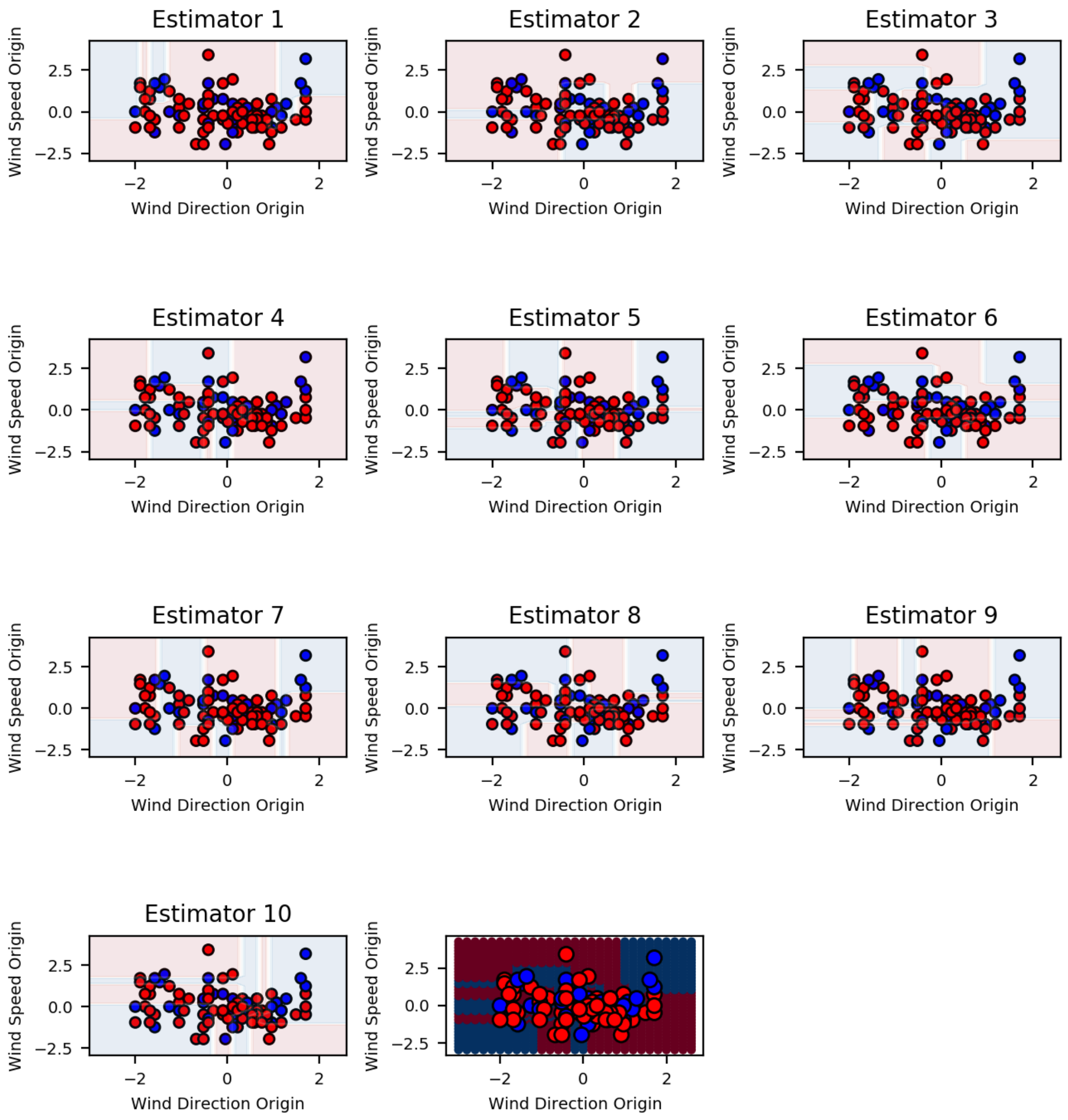


Figure 3.13: 10 Estimators and Final Decision Boundary for Adaboost

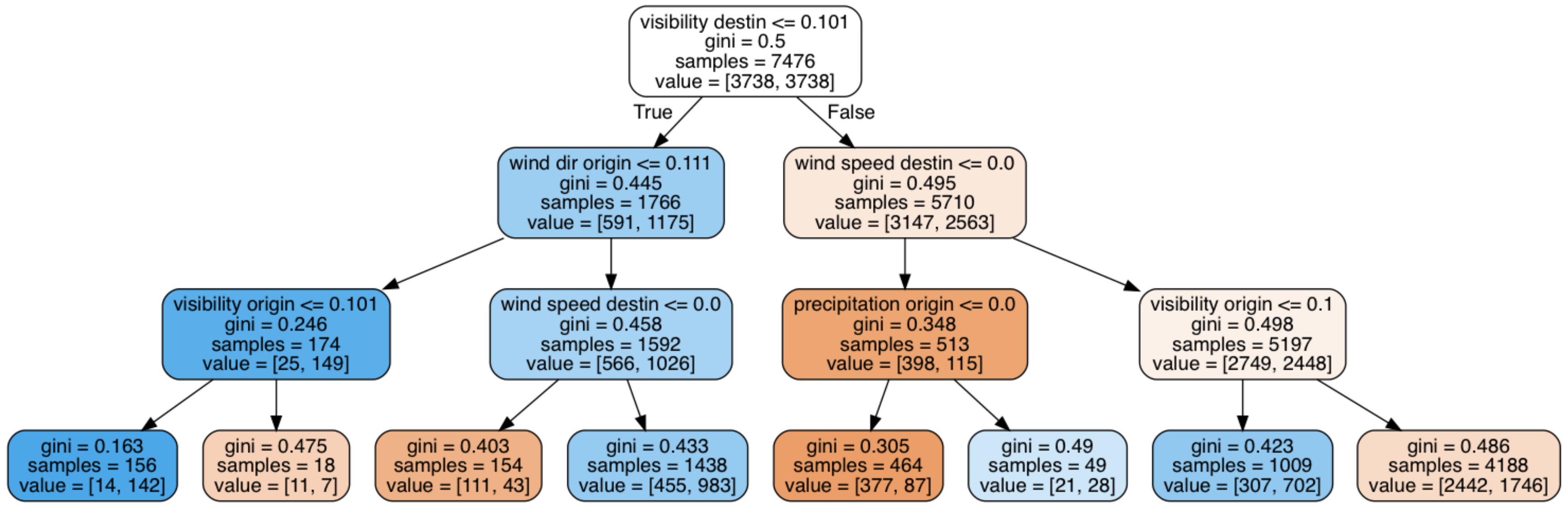


Figure 3.14: Typical Example of Random Forest Decision Rules

3.7.2 Threshold Selection

The classifier score value for each test point indicates how confident the classifier is in classifying the sample. For example, if a binary classifier yields a classifier score $[0.4, 0.6]$ for a sample, then the probability that this sample belongs to the positive class is 60%. On the other hand, the probability that the example is a negative class is 40%. When the threshold between the positive and the negative is 0.5, a sample is classified as the larger of the two classes. The threshold between the positive and the negative is often 0.5, but the threshold can be adjusted to optimize model performance. Higher thresholds mean more conservative classifier with higher precision, $\frac{TP}{TP+FP}$. This is because a positive class is predicted only when the classifier score is higher than the higher threshold. The performance metrics such as accuracy and AUC also depend on the threshold.

Figure 3.15 - Figure 3.17 is the probability density function of the classifier score for the

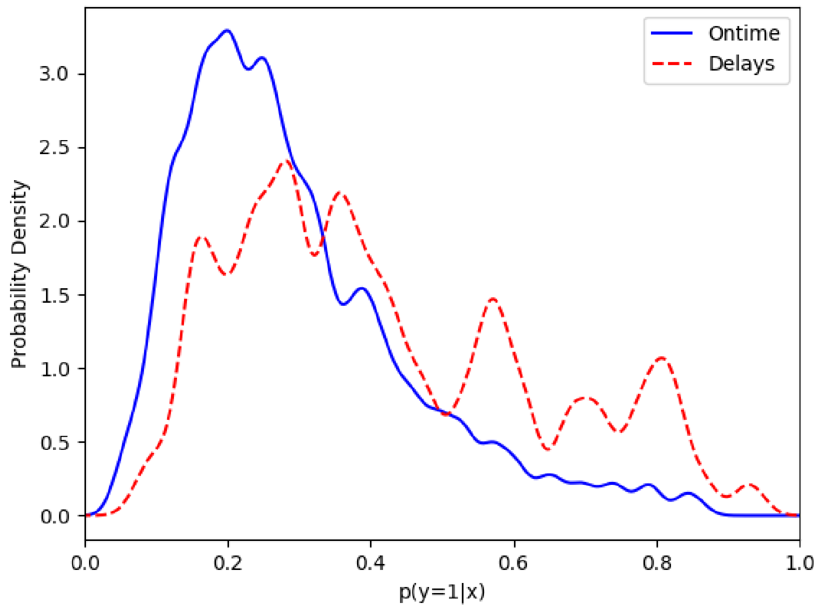


Figure 3.15: Probability Density Function of Random Forest for On-time and Delays

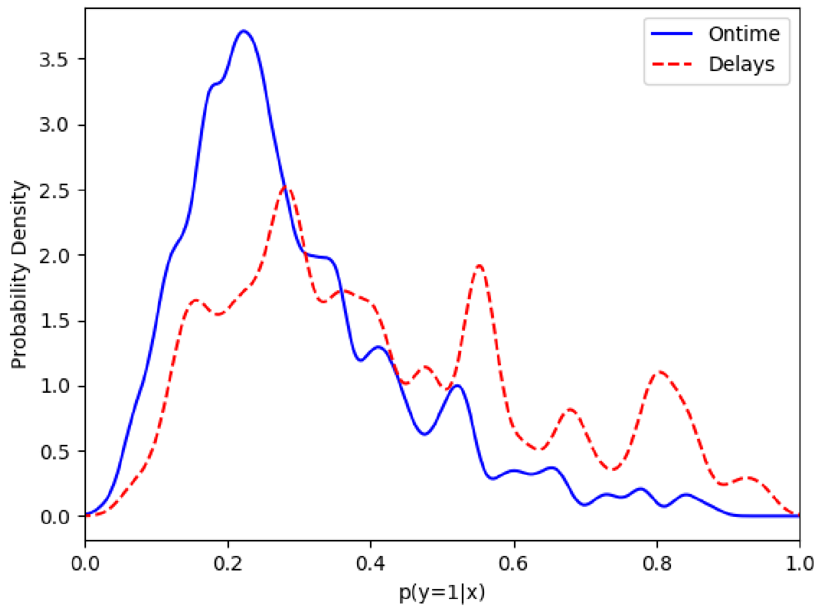


Figure 3.16: Probability Density Function of AdaBoost for On-time and Delays

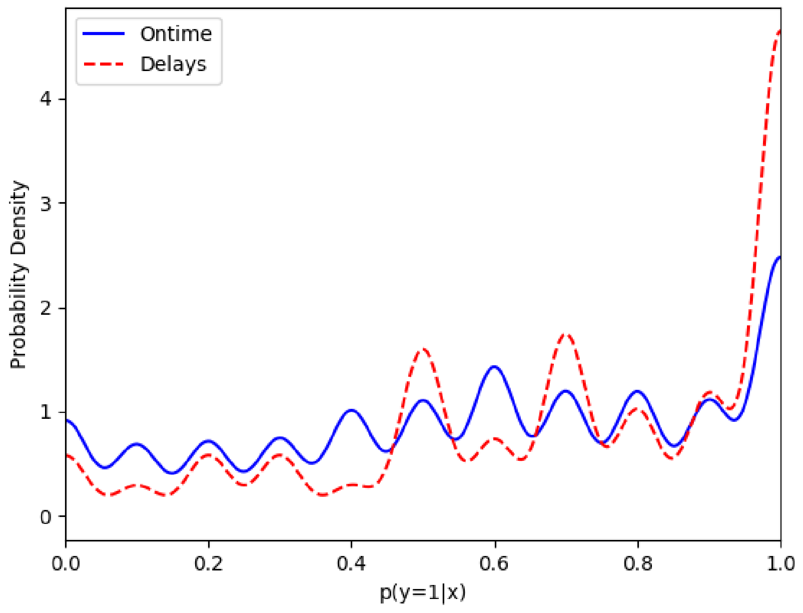


Figure 3.17: Probability Density Function of kNN for On-time and Delays

positive class. The samples of the test set are displayed in blue and red lines, respectively, according to their actual class. The x-axis is the probability that the model predicts a particular sample as the delayed class. The y-axis shows a relative likelihood of x. In Figure 3.15, if the threshold is set to 0.6 by drawing the line at $x = 0.6$, the left side of $x = 0.6$ in the blue line corresponds to TN, and the rest corresponds to FP. Since the dotted red line indicates samples that are the actual delayed, the left side is FN, and the right side is TP. Given that the on-time indicated by the blue solid line has a lot more samples, a threshold greater than 0.5 is expected to be favorable for higher accuracy. A threshold lower than 0.5 is expected to be advantageous to obtain a higher AUC.

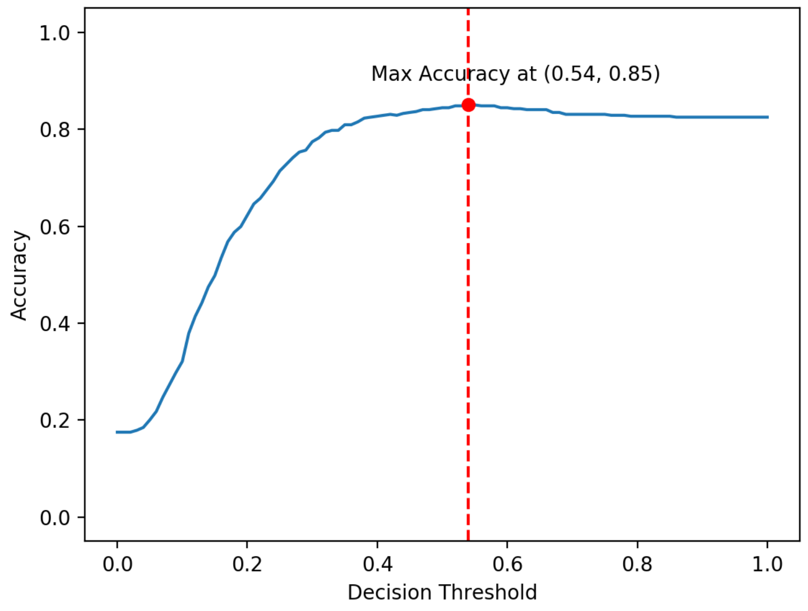


Figure 3.18: Threshold vs. Prediction Accuracy of Random Forest

AUC and accuracy changes according to the threshold were investigated for Random Forest. Figure 3.18 shows the prediction accuracy of Random Forest with a threshold value. The accuracy is measured for threshold values from 0 to 1. The accuracy gradually increased from a small value to a maximum when the threshold is 0.54. As the threshold increases, TN increased, and TP decreased in Figure 3.15. As a result, the accuracy, $\frac{TP+TN}{TP+TN+FP+FN}$, increased since the absolute number of TN is greater than TP. When the threshold value is above 0.5, the accuracy was saturated and maintained almost the same value.

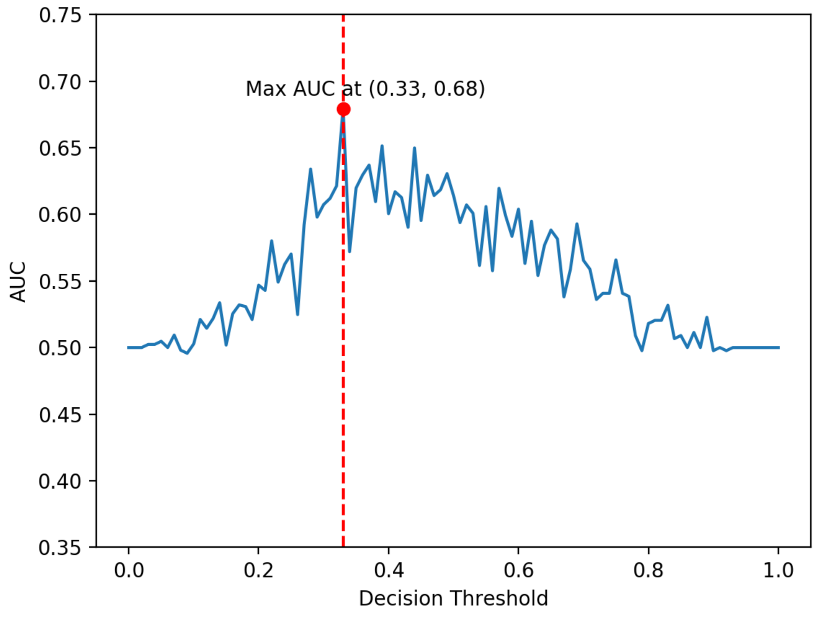


Figure 3.19: Threshold vs. AUC of Random Forest

As we did for the accuracy, AUC for the threshold from 0 to 1 is graphically displayed in Figure 3.19. The change in AUC according to the threshold change is different from that of accuracy. The AUC graph has a kind of bell shape that increases to a certain point and then decreases. The threshold for the maximum AUC is 0.33, which is considerably lower than the commonly used value of 0.5. In order to focus on predicting the minority class in the imbalanced dataset, it would be better to lower the threshold below 0.5. This gives you a better AUC, although at the expense of accuracy. Since the accuracy graph in Figure 3.18 is relatively flat for a threshold above 0.4, moving the threshold a little to the left for a higher AUC will not degrade the accuracy significantly.

3.7.3 Effects of Weather Data

First of all, the impacts of weather data on the prediction performance of the three models are investigated. In the series of graphs in Figure 3.20 - Figure 3.22, red dotted lines are from the model trained only with schedule data while blue solid lines are from the model trained with weather data as well as schedule data. A diagonal line represents random guessing, so the closer ROC curve to the diagonal, the less accurate.

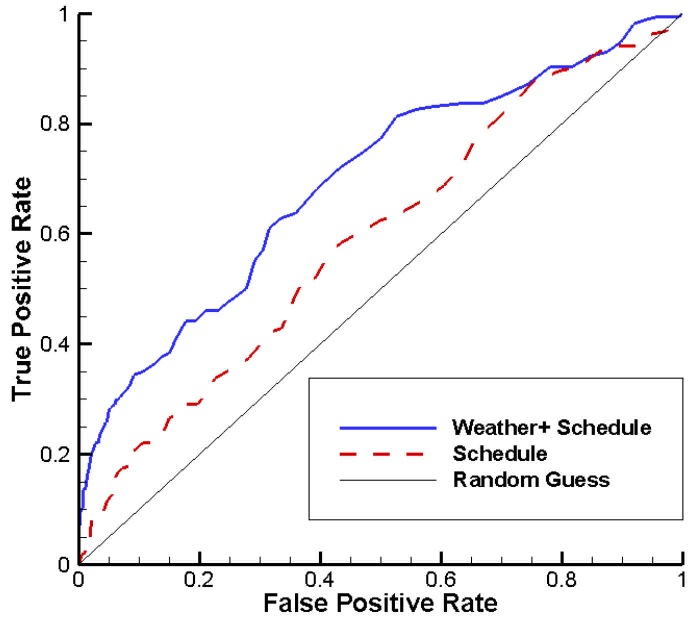


Figure 3.20: Receiver Operating Characteristic of Random Forest with and without Weather Data

In the case of Figure 3.20 and Figure 3.21, weather data distinctly improved predictive ability of each model since the solid blue lines are closer to the ideal point on the ROC curve than dotted red lines. Without weather data, the prediction performance of AdaBoost was about the same level as random guess'. In other words, AdaBoost could not tell whether a scheduled flight would be delayed or not without weather data. However, kNN's gap between two curves with and without weather data is quite small compared to the other

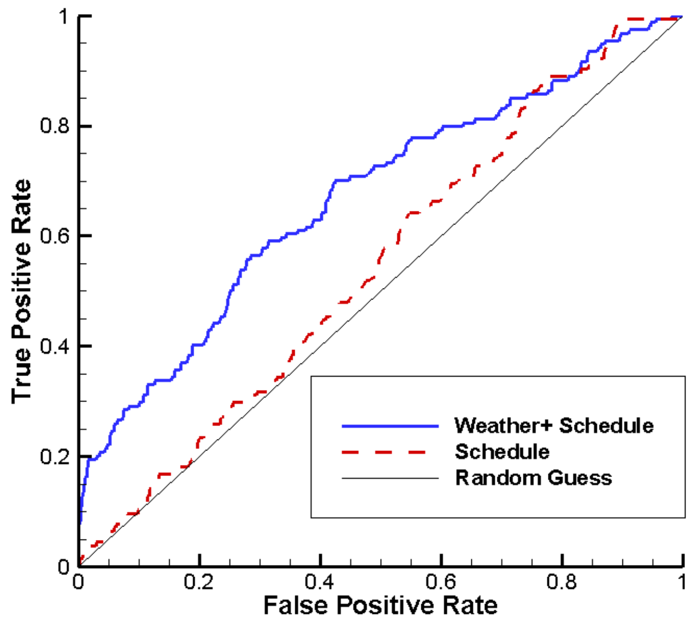


Figure 3.21: Receiver Operating Characteristic of AdaBoost with and without Weather Data

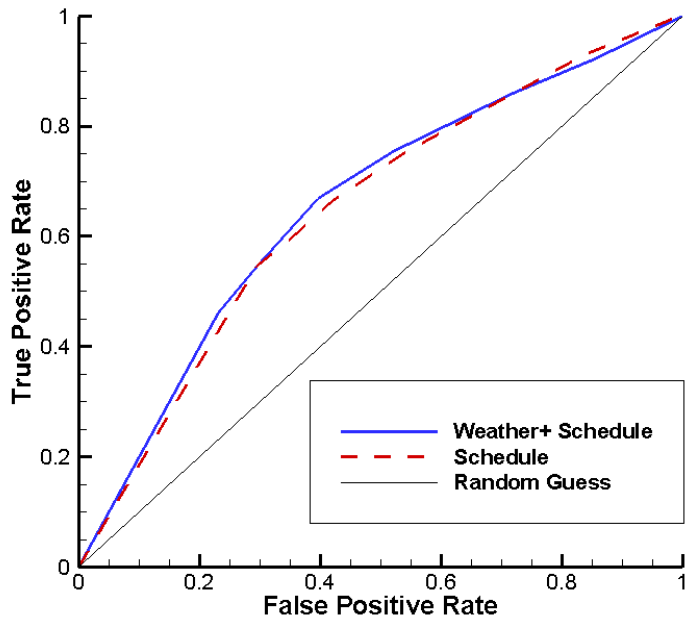


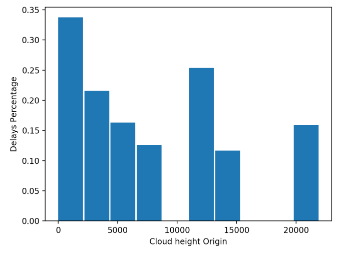
Figure 3.22: Receiver Operating Characteristic of kNN with and without Weather Data

classifiers. The reason for this is the curse of dimensionality. The number of features in the flight data reaches 56, and there are countless combinations of variables. In such high-dimensional data, measuring a distance between sample points becomes meaningless [155]. Moreover, adding weather data with 80 features did not help kNN to enhance predictive capability.

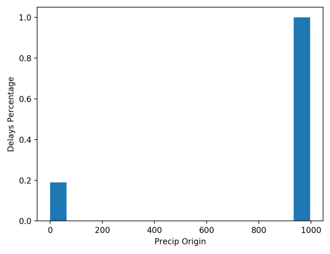
In addition to the effect of weather data, we also examined the effect of each weather variable on delays. The probability of occurrence of delays by origin and destination weather variables is shown in Figure 3.23 and Figure 3.24. The delays occurrence probability is calculated by counting the number of delays samples among the total number of samples belonging to the corresponding bin. Considering that the percentage of delayed flight among the entire scheduled flight is about 20%, bins with the probability greater than 20% contribute significantly to delays.

Heavy liquid precipitation, low ceiling height and visibility or high wind speed at the origin airport highly increased the probability of delays occurrence. This result is consistent with reality. In the case of heavy precipitation, it can be considered as a wet or contaminated runway [156] and special attention should be paid to the use of the runway at this time. Besides, airport runways cannot be operated when crosswind or tailwind speeds are greater than $15\,mph \approx 6.71\,m/s$ [157]. If the non-headwind blows faster than 15mph on a runway, the operating rate of the runway will decrease, which may affect punctuality. Regarding visibility, more restrictive rules are introduced for aircraft operation if the ceiling height and visibility do not meet the conditions listed in Appendix B. Restrictive IFR results in a lower operating rate compared to VFR [157].

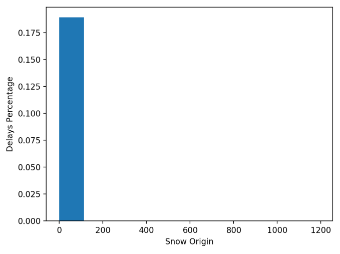
However, the variables that affected delays of the origin airport do not have a significant effect on the delays of the arrival airport. For example, even with low ceiling height and strong wind, the occurrence of delays did not stand out in Figure 3.24. The climate at the destination airport was different from the origin airport, so it did not have significant precipitation. However, the delays caused by heavy snowfall was clearly identified. The snow piled reduces surface friction coefficient of the runway surface. This increases the landing distance and affects aircraft directional control in crosswinds. This may reduce the operating rate of the runway and the runway may not be available at all in extreme case.



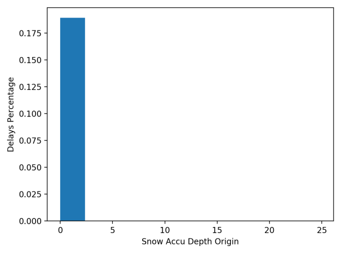
(a) Cloud Height $[m]$



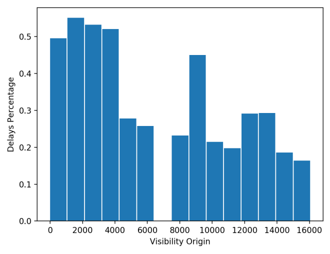
(b) Precipitation $[mm]$



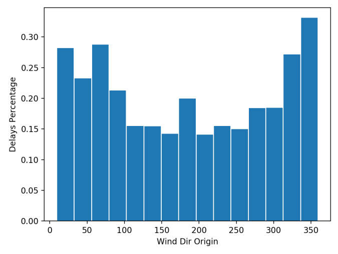
(c) Snow $[cm]$



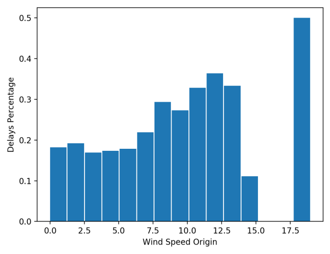
(d) Snow Accumulate Depth $[cm]$



(e) Visibility $[m]$

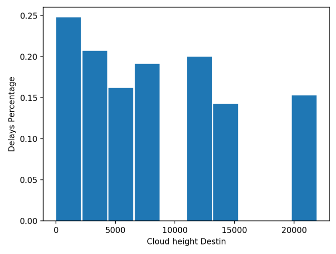


(f) Wind Direction $[deg]$

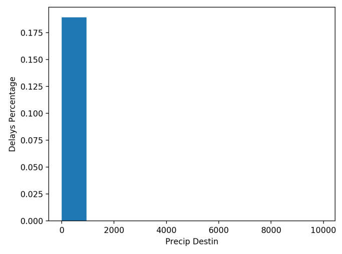


(g) Wind Speed $[m/s]$

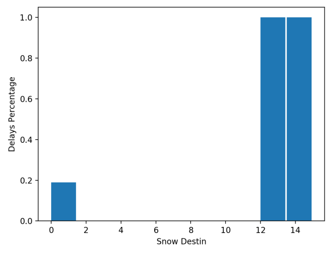
Figure 3.23: Histogram of Delays Occurrence Probability according to Origin Weather Variables



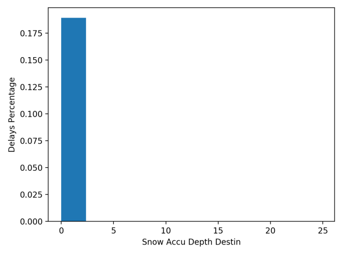
(a) Cloud Height $[m]$



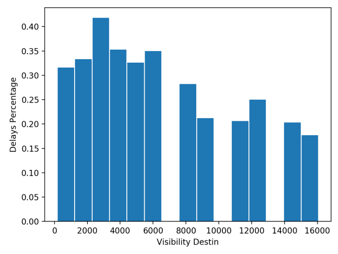
(b) Precipitation $[mm]$



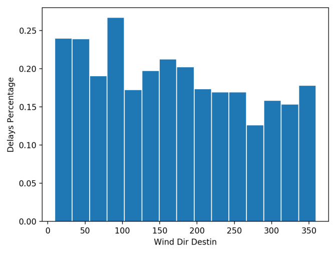
(c) Snow $[cm]$



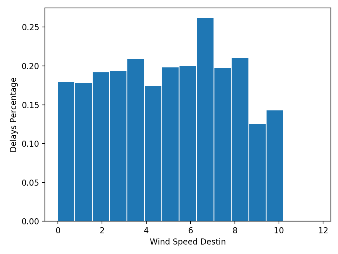
(d) Snow Accumulate Depth $[cm]$



(e) Visibility $[m]$



(f) Wind Direction $[deg]$



(g) Wind Speed $[m/s]$

Figure 3.24: Histogram of Delays Occurrence Probability according to Destination Weather Variables

3.7.4 Effects of Sampling Techniques

As it is already explained in Section 3.3, a combination of SMOTE and random under-sampling was utilized to balance the number of delayed samples against that of the on-time samples. To analyze the effects of sampling techniques, the model is trained with and without sampling techniques. Then the prediction performance of those two cases is compared. Table 3.6 shows the accuracy and the elapsed time of three classifiers trained with and without sampling techniques.

Table 3.6: Training Accuracy and Elapsed Time

(a) with Sampling Techniques

Classifier	Accuracy (%)	Elapsed Time(sec)
Random Forest	84.22	8
AdaBoost	84.24	12
kNN	61.69	1

(b) without Sampling Techniques

Classifier	Accuracy (%)	Elapsed Time(sec)
Random Forest	85.02	9
AdaBoost	84.44	12
kNN	81.12	1

Sampling techniques' influence on the predictive performance could be figured out in the comparison between Table 3.6(a) and 3.6(b). The accuracy of classifiers trained without sampling techniques was higher than those trained with sampling techniques. However, it does not imply that applying sampling techniques is a bad choice. The classifiers are biased toward the on-time class when they are trained with imbalanced data. Thus, it is easier for the classifiers to predict the on-time class. Further, it is more likely to classify the delayed flights as the on-time. Given that more than 70% of the total samples are on-time, models biased towards the on-time class will be more advantageous in terms of accuracy.

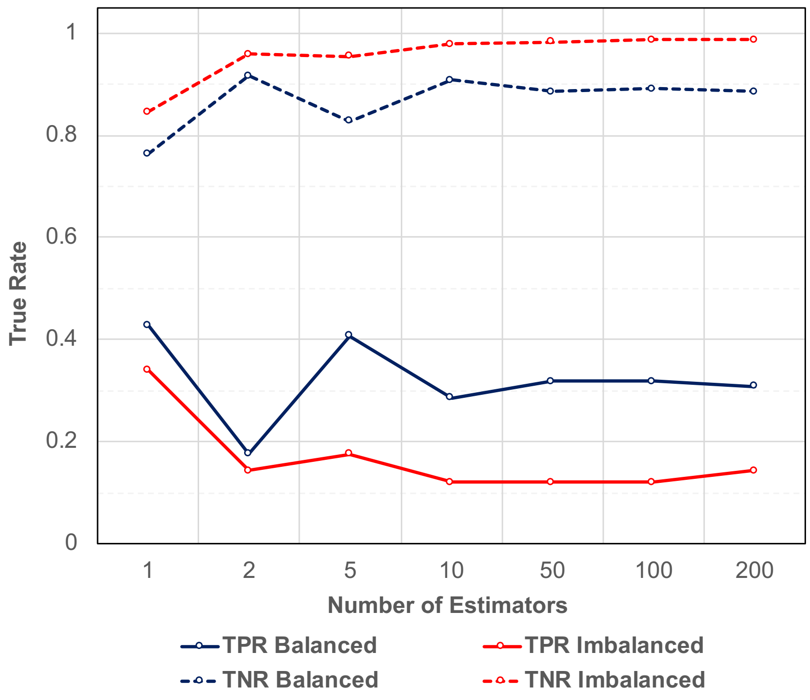


Figure 3.25: TPR and TNR of Random Forest according to Number of Estimators with and without SMOTE

TPR and TNR on the test data are compared in Figure 3.25 - Figure 3.27 to closely examine the effect of sampling on the performance of each classifier. TPR is the ratio of flights predicted to be delayed from the actually delayed flights. TNR represents the percentage of flights that are predicted to be the on-time class among the actual on-time flights. In the figures, TNR is represented by a dotted line and TPR is a solid line. The line color indicates whether the classifier was trained with the imbalanced raw data or with the balanced data where the number of samples between classes was adjusted by the sampling method.

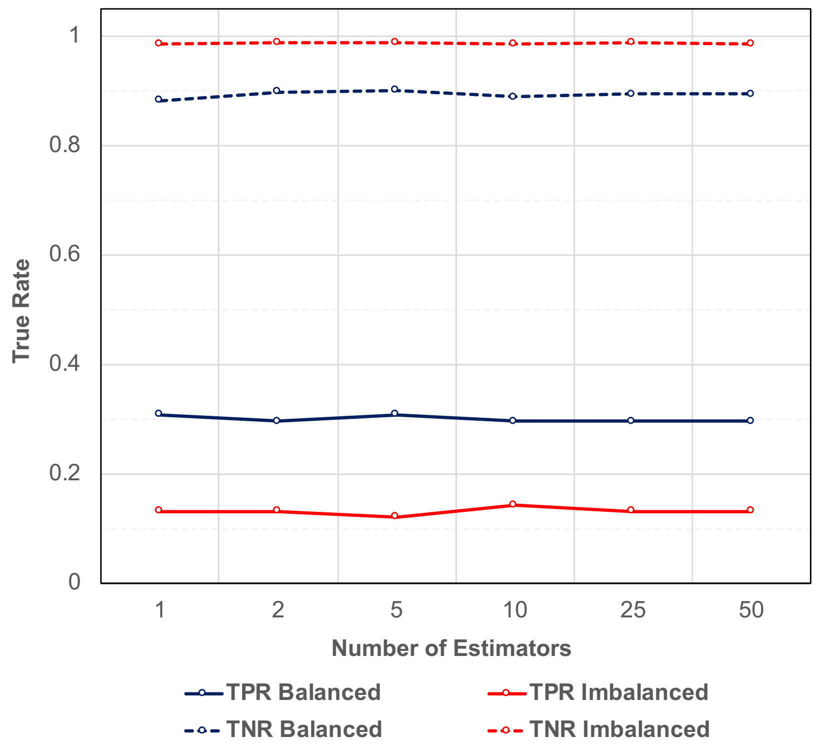


Figure 3.26: TPR and TNR of Adaboost according to Number of Estimators with and without SMOTE

In Figure 3.25 - Figure 3.27, the balanced data showed a higher TPR than the imbalanced for all data points, while TNR was higher for the imbalanced data. The significant changes in kNN's TPR and TNR after the application of the sampling method in Figure 3.27 is noteworthy. After balancing the data, kNN showed a significant increase in TPR and a significant decrease in FNR compared to the other two models. This can be interpreted to be due to the nature of kNN, which is heavily dependent on the information of a certain number of nearest neighbors [158]. That is, the sampling technique created neighbors that are delayed, and kNN became sensitive to delays.

The sampling technique lowered TNR and increased TPR. It means that the sampling technique improved the model's minority class recognition and the sensitivity of the model. It worked as intended with the introduction of SMOTE. Converting imbalanced data to

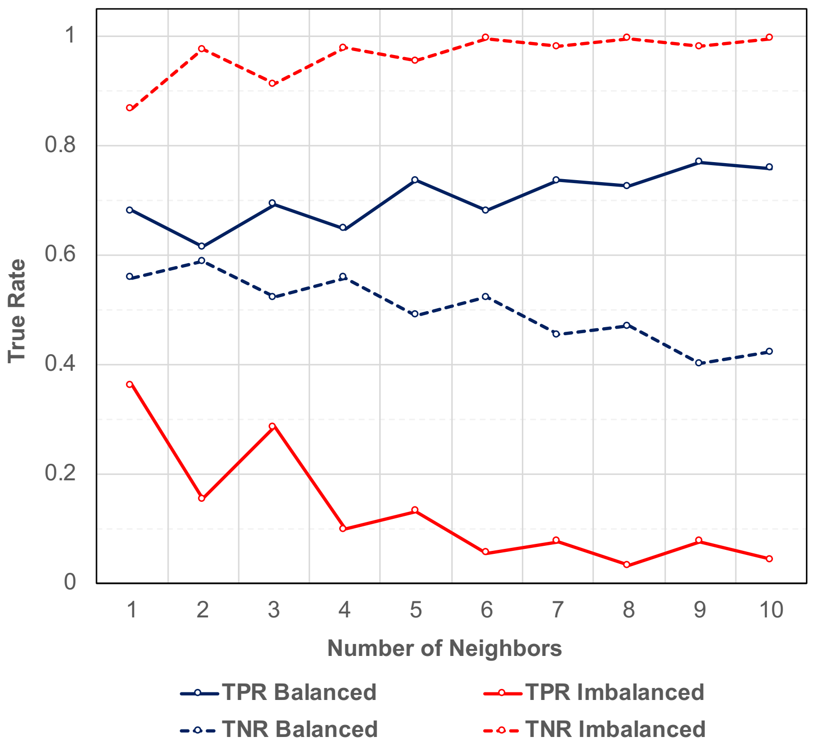


Figure 3.27: TPR and TNR of kNN according to Number of Estimators with and without SMOTE

balanced data resulted in a TPR gain greater than a TNR loss. This implies improved informedness of the model.

$$Informedness = TPR + TNR - 1$$

Informedness quantifies how informed decisions a classifier has made [153]. Informedness is an evaluation criterion that regards TPR and TNR as equal weights. Since the positive class in the flight delays prediction problem may require a higher cost than the negative class, TPR and TNR with appropriate weights instead of 1 will be discussed in Chapter 4.

Despite the increase in TNR with sampling technology, there is still room for improvement. False positive rate (FPR) is the proportion of the on-time flights that are incorrectly classified and is equivalent to $(1 - TNR)$. The percentage of delayed flights that are classified as the on-time, $(1 - TPR)$, is false negative rate (FNR). Comparison between FPR and FNR revealed that the models are more likely to misclassify the delayed class as the on-time class and a considerable number of delayed flights were not able to be captured by the model. This is because the model fitted in this study didn't take into account other possible causes of delays than weather. If delays are occurred due to the fact that is not weather, it is hard to be predicted by the model. For example, in case of a flight arrived late because of congestion in air traffic, the model might not recognize it since it is nothing to do with weather.

3.7.5 Assessment of Performance on Test Data

In addition to cross-validation, the performance of the classifier should be validated for data that has never been used in training of the classifier. For the purpose, the classifiers are get tested with a test set consists of 56 flights departing from DEN and arriving at CLT during a week, May 20^{th} to 26^{th}, 2016. The test performance of the classifiers is shown in Table 3.7.

Table 3.7: Test Accuracy and Width of Decrease from Training Accuracy

Classifier	Accuracy (%)	Decrease (%)
Random Forest	80.36	3.86
AdaBoost	71.43	12.81
KNN	35.71	25.98

Table 3.6 shows the predictive performance of Random Forest, AdaBoost, and kNN for the test set. The last column of the table is a width of decrease from the training accuracy. The classifiers performed better on the training data and predicting unseen data was harder for them. This is because the training data is a subset of all the data and can not represent the entire data. If we have the whole data, an exact classification function can

be obtained. However, since the classification function is estimated only from a subset of data, the generalization performance for the test data may be less than the training data.

The ultimate goal of the model is to predict delays of scheduled flights not yet flown with the weather forecast. Hence, Random Forests' behavior with five-day and one-day forecast horizon was assessed. The test results are in Figure 3.28. The first and second bar in Figure 3.28 are prediction results obtained with five-day forecast horizon and one-day forecast horizon, respectively. The last bar of Figure 3.28 is the results from actual weather.

In case of the results with the forecast, there could possibly be the error arose from the forecast uncertainty combined with model's pure error. Classification results attained from the actual weather was required to differentiate uncertainty in the forecast from the model's pure error. In Figure 3.28, the predictions with the forecast were much worse than the predictions with the actual weather. The model's predictive performance is drastically lowered due to uncertainty in the forecast. The results with the actual weather exhibited higher accuracy as uncertainty in forecast had dropped out. The weather forecast uncertainty will be quantified in Section 3.7.6.

3.7.6 Adding Weather Forecast Uncertainty to the Models

This section explores changes in model performance due to uncertainty in the weather forecast. There are two types of uncertainties in the weather forecast: Epistemic and Aleatory. Epistemic uncertainty represents the uncertainty of a model due to lack of knowledge or information. Epistemic uncertainty can be reduced by introducing additional information [159]. The epistemic uncertainty of the weather forecast is determined by how precisely the model is built. Aleatory uncertainty refers to the inherent variability in the system under consideration [160]. Unlike epistemic uncertainty, this type of uncertainty cannot be eliminated because the variability is always present in the system. The source of the aleatory uncertainty in the weather forecast is unexpectedness of weather condition. Even with a sophisticated forecast model, forecasting the future, especially in the distant future, cannot

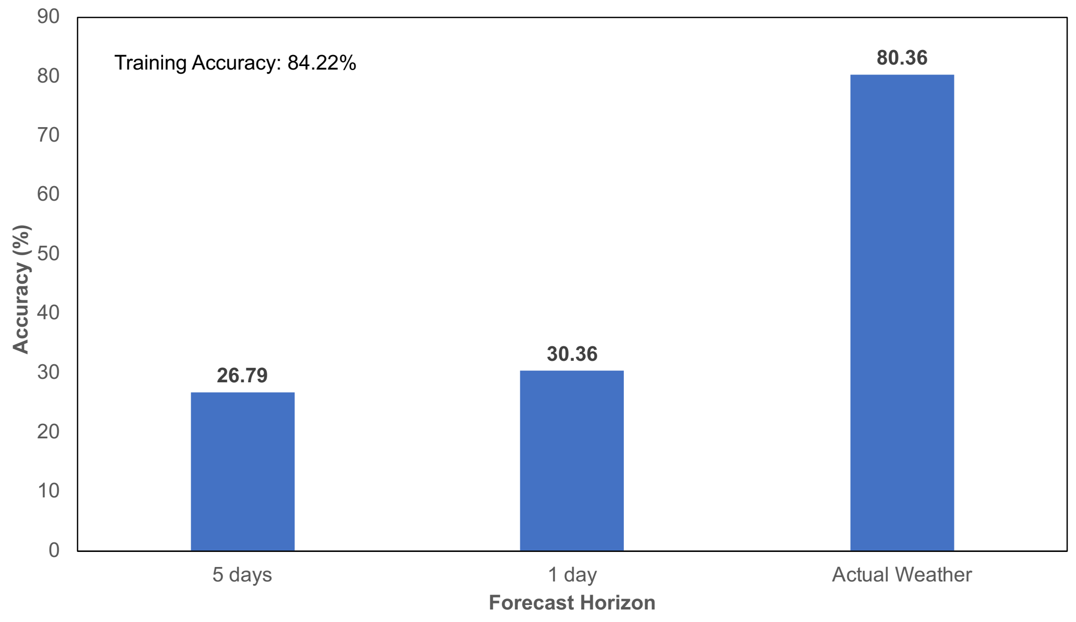


Figure 3.28: Test Accuracy of Random Forest with Forecast Horizons

be perfect due to the aleatory uncertainty.

In the forecast problem, the term 'horizon' refers to how far forward the forecast is for the future. The degree of uncertainty varies depending on the width of the horizon, such as ten-day, one-day, or one-hour. Intuitively, it is more confident when predicting the near future than forecasting the distant future. In order to analyze this trend that was seen in Figure 3.28, we first have to determine the distribution of the weather forecast. Next, we define the plausible range that each variable can have. Then the uncertainty can be quantified by the sensitivity of the result to a variable of a given interval [161].

The error distribution of weather forecast can be estimated through the distribution of historical weather data. Figure 3.29 shows the distribution of each weather variable through boxplots. The distribution of normalized values is different for each variable. Variables such as wind direction or cloud height have high variability. On the other hand, for the other variables, samples are concentrated at low values, and the variability is quite small.

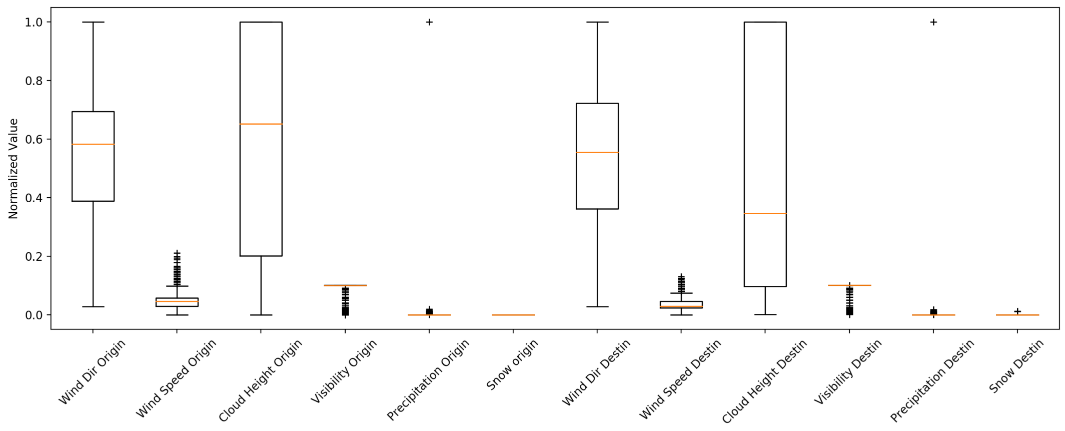


Figure 3.29: Boxplot for Data Distribution according to Weather Variables

Let us look at how to estimate the statistic for the weather forecast from the statistics of the weather data included in the training data. Note that the estimation is from a finite sample not the entire population for the weather data. Therefore, the statistics for the sample dataset are as follows [162]. The sample mean is

$$\bar{X} = \sum_{i=1}^{n} \frac{x_i}{n}$$

where

$X \subset x_1, x_2, \ldots$ is a random variable,

N is the number of the entire population,

n is the number of a finite sample.

The sample variance is

$$s^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{X})^2}{n-1}. \tag{3.1}$$

We know that the equation for computing the actual variance is

$$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N} \tag{3.2}$$

where μ is the actual mean. Compared Equation 3.1 and Equation 3.2, the reason why the denominator of Equation 3.1 is $(n-1)$ rather than N is that the denominator is the number of degrees of freedom, not the number of sample data [162]. It is called Bessel's correction. $(n-1)$ can correct the bias that occurs when calculating the squared deviation using the sample mean instead of the actual mean. Then, the sample standard deviation is

$$s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{X})^2}{n-1}}.$$

The Central Limit Theorem states that when the independent random variables are added, the sum tends to be a normal distribution regardless of the original distribution. If we consider the error as a sum of many random variables, the weather forecast error of each variable can be simulated as a normal distribution. Therefore, it is possible to test the impacts of the weather forecast error on the classifier by estimating the confidence interval of the normal distribution for each variable. When the range of each variable is the confidence interval, the change of the classifier is the impacts of the uncertainty of the weather forecast to the classifier.

Assuming that the mean of the distribution is the same as the actual value, the remaining is to estimate the standard error of forecast. Under the mean forecast assumption, the estimated variance of the forecast error is the sum of the estimated variance of intrinsic risk and the estimated variance of parameter risk [162]. The estimated variance of intrinsic risk can be expressed as a sample variance, s^2, which measures the noise of the data. The remaining term, the estimated variance of parameter risk is the error in estimating the signal in the data. This is equal to the sample variance divided by the number of samples. The standard error of the forecast, $SE_{forecast}$, is the square root of the estimated variance. Thus, $SE_{forecast}$ can be expressed as follows:

$$SE_{forecast} = \sqrt{s^2 + SE_{mean}^2} = \sqrt{s^2 + \frac{s^2}{n}} = s\sqrt{1 + \frac{1}{n}}$$

$$\approx s\left(1 + \frac{1}{2n}\right).$$

In our case, the number of samples is very large, so the standard error of the forecast can be approximated as the sample standard deviation by the above equation:

$$SE_{forecast} \approx s \text{ (when n is very large)} . \tag{3.3}$$

The standard error of the forecast for each weather variable obtained by Equation 3.3 is presented in Table 3.8. The confidence interval is the range of a parameter so that the parameter's value is within the interval with a particular probability. Assuming that the prediction error follows a normal distribution, the confidence interval for 95$\%$ confidence level is constructed to be deviated from the central value by $\pm 2 \cdot (SE_{forecast})$. Since $SE_{forecast}$ is proportional to the sample standard deviation, the variables with high variability in Figure 3.29 have the wider confidence interval for $SE_{forecast}$.

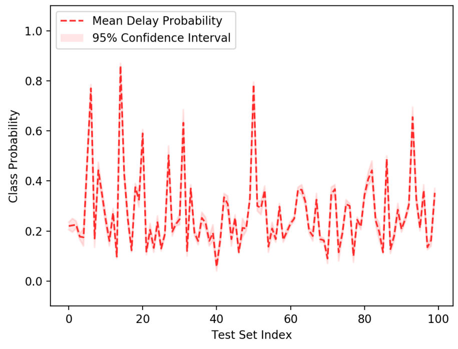
The propagation of the weather forecast uncertainty for the delays prediction model is shown in Figure 3.30 and Figure 3.31. 100 sample points were selected to construct a test set and listed in the x-axis direction. Also, Individual assessments of each weather variable were made with the values of the other variables fixed. The class probability is used for measuring propagated weather forecast uncertainty to the delays prediction model, instead of binary delays variable. The class probability is the model's level of confidence that a sample is delayed. The weather forecast uncertainty formed the interval of the class probability. The graph for snow at the origin is omitted because $SE_{forecast}$ for snow at

Table 3.8: The Estimated Standard Error for Each Weather Forecast

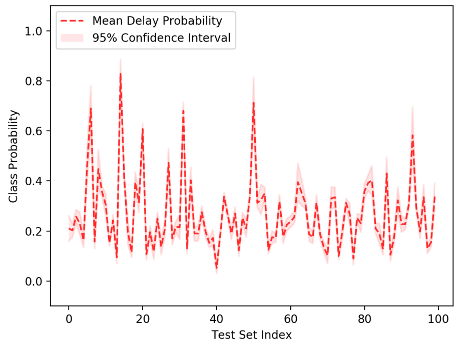
Origin Weather	$SE_{forecast}$	Destination Weather	$SE_{forecast}$
Wind Direction	0.2453	Wind Direction	0.2734
Wind Speed	0.0265	Wind Speed	0.0207
Cloud Height	0.4120	Cloud Height	0.4240
Visibility	0.0226	Visibility	0.0185
Precipitation	0.0140	Precipitation	0.040
Snow	0	Snow	0.0002

the origin is 0. For a sample x, if the interval of the class probability spans the decision threshold, the sample may be classified as the delayed or the on-time.

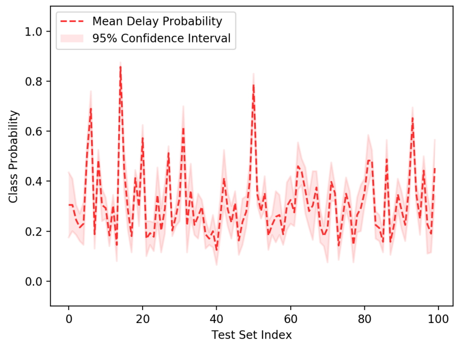
In Figure 3.30 and Figure 3.31, the width of interval for the class probability is different for each weather variable. This difference reflects the impact of the corresponding meteorological variable's uncertainty on the decision of the model. The wind direction at the origin, the wind direction at the destination, and the cloud height at the origin did not have a significant effect on the system response even though the standard error of the forecast was estimated to be a large value. The factors that amplify the uncertainty of the system response were the standard errors of the visibility at the origin, the cloud height at the destination, the precipitation at the origin, and the precipitation at the destination.



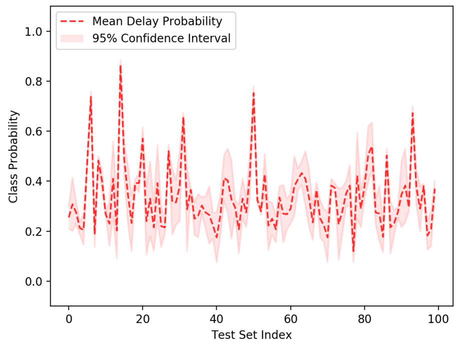
(a) Wind Direction



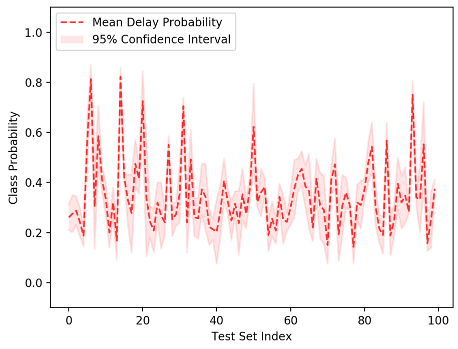
(b) Wind Speed



(c) Cloud Height

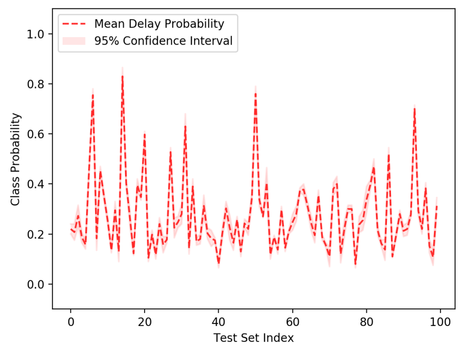


(d) Visibility

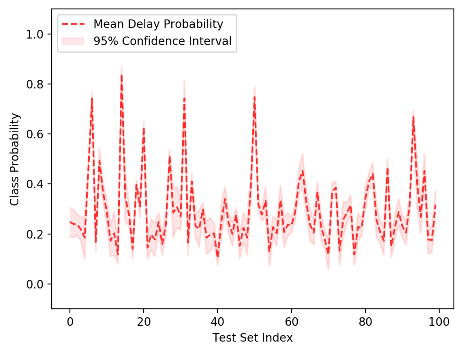


(e) Precipitation

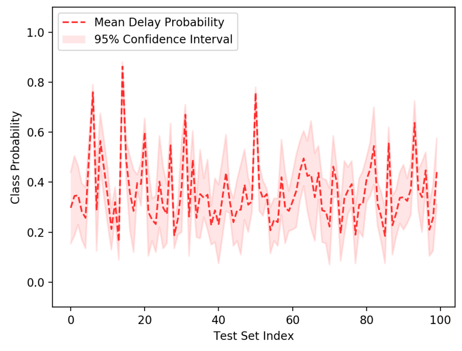
Figure 3.30: Quantification of Uncertainty in Origin Airport Weather Forecast for Class Probability



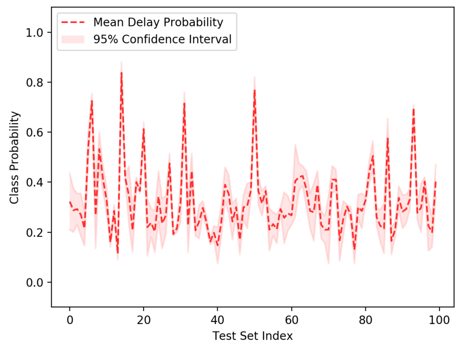
(a) Wind Direction



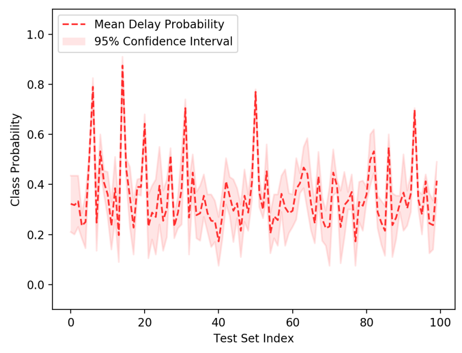
(b) Wind Speed



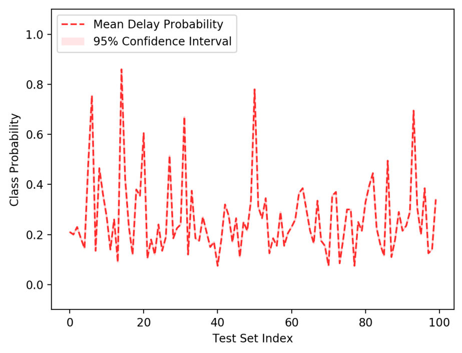
(c) Cloud Height



(d) Visibility



(e) Precipitation



(f) Snow

Figure 3.31: Quantification of Uncertainty in Destination Airport Weather Forecast for Class Probability

3.8 Chapter Summary

Chapter 3 describes the classification of individual flights delays using machine learning algorithms. Based on the observation that flight delays are affected by flight schedules and inclement weather conditions, the historical weather and traffic data from individual Origin-Destination (OD) pairs had been collected. The number of on-time flights within the dataset is significantly higher than the number of delayed flights, which could potentially bias the model toward the on-time flights. To resolve this problem, a combination of SMOTE and random under-sampling had been introduced. The impact of weather data and the impact of the sampling method has been revealed. The classifiers were trained with balanced data through the sampling technique, and the algorithms implemented in this study includes Random Forests, AdaBoost and k-Nearest-Neighbors.

In this chapter, we have hypothesized that models trained with supervised machine learning algorithms can provide interpretability and imbalance data handling, but they are similar in accuracy to existing models. Several experiments had been conducted to demonstrate this. The performance of the classifier was optimized by adjusting the hyperparameters and thresholds for each algorithm. A method to help users understand the decision of the classifier using the decision plane and the tree structure has also been described. The model's prediction performance on the validation set and the test set was analyzed. With the help of the weather data and the sampling technique, we were able to predict the delays of individual flights successfully. In addition, the statistics of weather forecasts were estimated from the training data, and the influence of the weather forecast uncertainty on the model was analyzed.

CHAPTER 4
COST-SENSITIVE PREDICTION OF DELAYS

When predicting the on-time performance of flights, the delayed class is rare in the dataset, but the cost of not recognizing the examples belonging to the delayed class is high. Studies so far have simply focused on a model's prediction accuracy, and there have been few studies on the delays prediction problem taking into account misclassification costs. More research is needed to consider misclassification costs when predicting individual flight's delays.

4.1 Problem Definition

Traditionally, a performance of the most classification algorithm is measured based on accuracy. Accuracy is calculated by counting the number of correct classification:

$$\text{Accuracy} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{I}(y_i = \hat{y}_i)$$

where $\mathbf{I}(\cdot)$ is an function that returns 1 if $\cdot$ is true. y_i is the actual value of the i^{th} sample and $\hat{y}_i$ is the predicted value of the model for that sample. Conversely, an error is measured as the number of misclassified among the entire sample:

$$\text{Error} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{I}(y_i \neq \hat{y}_i).$$

Expressing a model's performance in this way is based on the assumption that all kinds of errors result in the same cost. However, in the delays prediction problem, the misclassification cost of the delayed class and that of the on-time class are considerably different. Therefore, a new model should be created to reflect this, and evaluation criteria other than the accuracy of the model should be established. The problem of creating a classifier that takes into account the imbalanced misclassification cost has the following characteristics.

- Imbalanced Data: Imbalanced data is the data where the classification categories are not approximately equally represented [163]. In the delays prediction problem, the delayed class occupies only 20% of the dataset, but the misclassification cost of this class is much larger than the on-time class as in Figure 4.1. The classifiers trained with the imbalanced data will be biased toward the on-time class and will cause the greater loss because expensive minorities are incorrectly predicted.

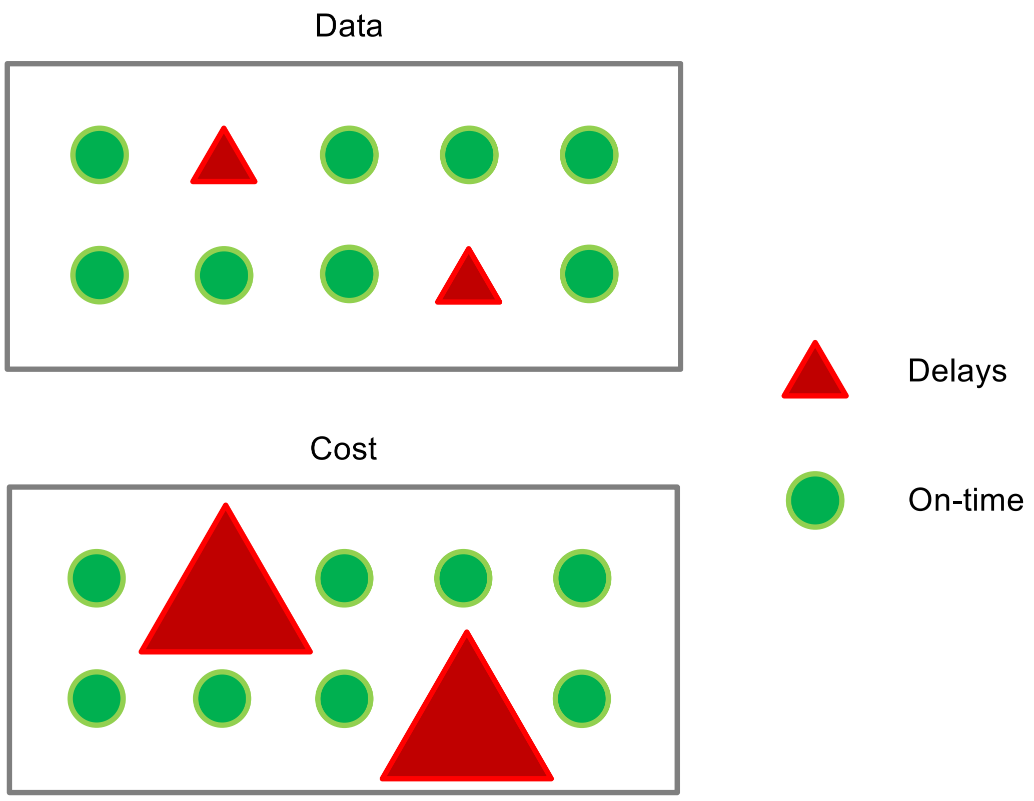


Figure 4.1: Imbalanced Flight Delays Data

- Unknown Misclassification Cost: The exact amount of misclassification cost is not known. From the previous studies, we know that the false negative error is more expensive than the false positive error but not the exact values.

By focusing on those facts, the third research question is introduced.

Research Question 3

When forecasting the on-time performance of flights, what is an appropriate method to handle asymmetric misclassification costs and minimize the total cost from incorrect prediction?

The goal is to train a cost-sensitive classifier that can predict individual flight's delays. The scope has narrowed down to US domestic flights operated by major airlines. We aim to predict arrival delays specifically. Departure and en-route delays minutes are captured at the end of flight by arrival delay minutes. To increase the predictive capability, data from a specific origin-destination (OD) pair is used to train models not from the entire National Airspace System (NAS) by capturing each pairs characteristics. The output from the trained model is either 0 (on-time) or 1 (delay).

4.1.1 Benefits from Correct Delays Predictions

If you can predict the delays a few hours ahead of the flight, the system will have the potential to improve the performance [164]. By taking the benefits of the accurate delays prediction described in Section 1.5.2, the cost of misclassification for delays can be explored indirectly.

Informing passengers of the possibility of delays in advance is giving them the ability to prepare for wait times. Passengers will know the delay information in advance so that they will gain some sort of control over their travel, instead of just waiting for an unspecified time at an uncomfortable airport gate. It helps them to minimize the impact of flight delays because they can actively manage the impact of interruptions. For example, passengers can sit comfortably at their home or reschedule the connection by contacting the airline at home.

The benefits of delay notification are greater who need special attention to flight. When a flight delay occurs, parents who fly with their babies may be in trouble due to lack of baby food or diapers. If the parents knew the delay in advance, they could prepare extra. Likewise, delays notification means a lot to patients who need special care. Depending on the presence of delays notification, the flight to them can be either relatively quiet or an emergency. Business travelers face economic problems when a flight is delayed. With the help of delays notification, they can reduce their economic losses by finding alternatives, such as being assigned to another flight, instead of missing important meetings totally.

Passengers are only concerned about delays on certain individual flights, but accurate delay notifications can lead to network-wide benefits for airlines. By knowing the delays information in advance, the airline can identify the cause of the flight interruption and investigate whether the cause can be eliminated. It will minimize the delays of the current flight. If the delays can be understood early, the airline can not only take action on the current flight but also stop the propagation of delays. All types of the system disturbances are likely to affect subsequent flights of the airline. However, the notifications can prevent these disturbances from propagating to the system.

In addition, the delay notification brings miscellaneous benefits. If there is a delay, the airline will receive numerous phone calls and online requests from passengers, and passengers will form a long line at the ticket counter. In turn, they make the ground operations of the airline inefficient. From the delays notification, the airline can reassign passengers or provide information to passengers in advance, thereby eliminating factors that have adverse impacts on the ground operation. Besides, the ATC is able to find preventive measures for the following flights, and the airline can establish a recovery plan.

4.2 Backgrounds

Although it has never been applied to flight delays prediction, cost-sensitive learning has been an important research subject in other areas with asymmetric misclassification cost. Joo et al.'s study is a typical example. They investigated the asymmetric costs of the false positive and the false negative to enhance intrusion detection systems (IDS) performance [165]. The purpose of IDS is to identify network intruders from normal users. A false negative error occurs when an IDS cannot detect an attack. This means that the attack will succeed and damage the target resource. A false positive error, on the other hand, is to classify regular users on the network as attacks. Therefore, the risk of false negative errors is higher than that of false positive errors. They utilized the neural network model to consider the cost ratio of false negative errors to false positive errors. As a result, the total cost decreased as the cost ratio of false negative and false positive errors increase.

Another important example of a cost-sensitive classifier is from the biomedical area. Yang et al. presented a modified random forest classifier that can be used in medical diagnosis [166]. Uguroglu et al. investigated classical machine learning algorithms for the diagnosis of heart disease [167]. In medical diagnosis, a false negative error may have more severe consequence than a false positive error since predicting a disease as healthy can be fatal. However, it does not mean that false positive errors are not costly. If a healthy person is diagnosed as having a disease, the patient will be charged for unnecessary care and test. Also, that treatment may worsen the patient's health [167]. The cost-sensitive classifier is developed to address this problem, and they could minimize the risk of misclassification.

4.2.1 Possible Improvements with sampling methods

Many classifiers including random forest, k-nearest neighbors and AdaBoost try to maximize prediction accuracy. They assume that the incorrect prediction of a sample will always incur the same cost. However, this is not true in many cases. In our case, predicting delays as on-time is more expensive than predicting on-time as delays. Thus, classifiers that do not consider costs would not yield practical results.

There are three general ways to handle the applicable asymmetric misclassification costs, regardless of the classification algorithm. The first one is called MetaCost. This method is to relabel the training samples to their estimated minimal-cost class [169]. The second one is to modify the learning algorithm itself. It tailors a training dataset to put relative costs as a new feature [50]. Then it adjusts the traditional training algorithms to use costs as a weight of a sample. The last one, sampling method, is changing imbalanced data into balanced data by subsampling. It carefully subsamples examples from the original dataset according to their relative costs and uses the new dataset for training [50]. By utilizing the new dataset, any cost-insensitive classifiers can be transformed into cost-sensitive one.

Table 4.1: Comparison of Cost-sensitive Learning Methods

Criteria	MetaCost [169]	Modifying Learning Algorithm [50]	Sampling [50]
Simplicity of method	×	△	○
Has theoretical guarantee?	×	○	○
Computational Cost	×	○	○

Table 4.1 compares the three methods. In this research, the sampling method is chosen for the following strength [50]. Firstly, it has theoretical guarantees that its performance is no worse than traditional learning. Moreover, its performance has proven with publically available datasets. Also, it drastically reduces the computation which is required by other methods.

The elaboration of cost-proportionate rejection sampling enables the formulation of the third hypothesis.

Hypothesis 3

If the cost-proportionate rejection sampling method is applied to the prediction model, then the model can find the optimal prediction with the lowest misclassification cost.

4.3 Methods

The definition of cost-sensitive learning is a type of learning that takes the misclassification costs into account [168]. The goal of cost-sensitive learning is quite different from the general learning algorithms. It tries to minimize the total cost of misclassification while general algorithm minimizes the number of misclassification. The key difference between cost-sensitive learning and cost-insensitive learning is that cost sensitive learning treats different misclassifications differently. The essense of cost-sensitive learning is that it may be best to act as if the delays class is true even when the on-time class is more likely. The decision of the classifier depends on the cost when the sample is misclassified.

4.3.1 Rejection Sampling

The concept of this method can be easily explained by Figure 4.2. The black line shows the original distribution, $D(x)$, and the red line is the target distribution, $\hat(D)(x)$. The goal of this algorithm is to make the data with the original distribution, $D(x)$, follow the target distribution, $\hat(D)(x)$. Firstly, an x-position is sampled from the original density function. Then, a vertical line is drawn from this x position until it touches the curve of the original distribution. The final step determines whether to accept the sample, x, or not. To do that, a random number is drawn from the uniform distribution ranging from zero to the value of the original distribution at x. If the random number is greater than the value of the target distribution, $\hat{D}$, at x, the point x is rejected. A new dataset that follows the target distribution can be obtained by repeating this process for the original data.

4.3.2 Cost-proportionate Rejection Sampling (Costing)

The cost-proportionate rejection sampling with aggregation (called costing) proposed by Zadrozny et al. has achieved excellent predictive performance and drastic savings of computational resources [50]. The costing method can minimize the misclassification cost of

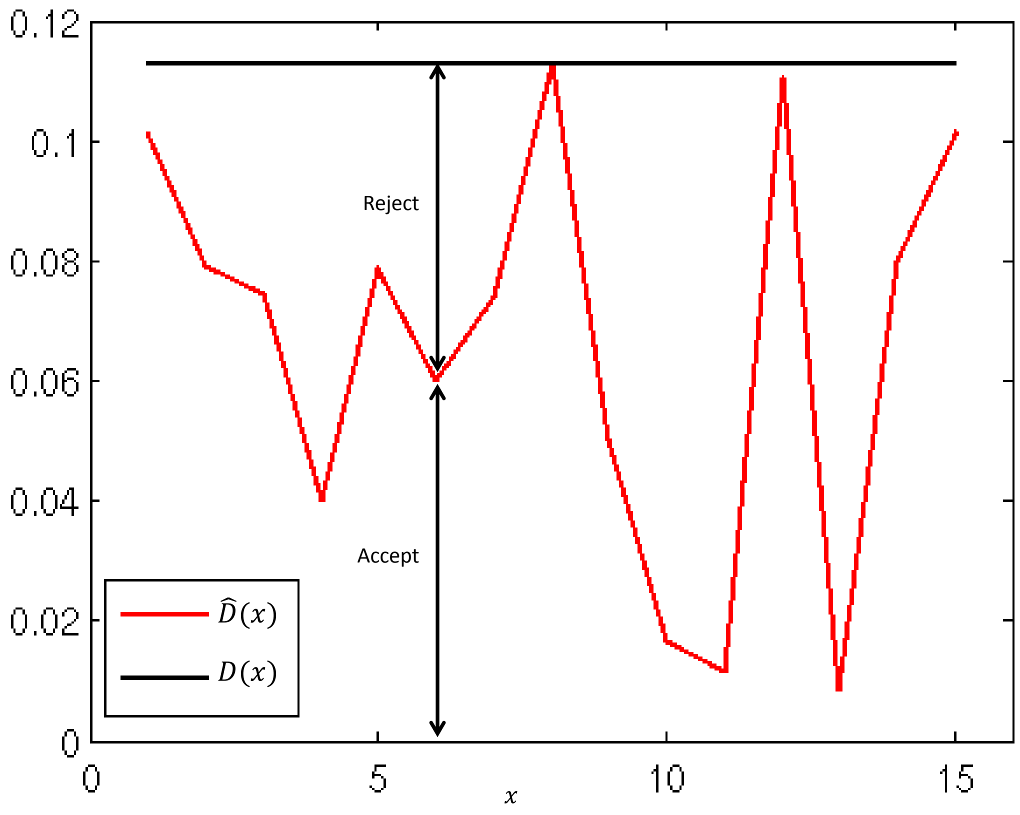


Figure 4.2: Schematic Drawing of Rejection Sampling with Target and Original Density Function

any cost-insensitive classifier. It alters the original distribution, D, to another distribution $\hat{D}$, by multiplying by a factor that is proportionate to the relative cost of each example. Returning to Figure 4.2, the point where the red line and the vertical line meet is the relative cost of a sample. The acceptance of a sample is determined by the length under the red line. If it is an expensive sample, it will have a high probability to be included in the new dataset.

In Algorithm. 3, the costing algorithm is explained step by step. At first, a sample $(\mathbf{x}, y)$ is drawn from S where $\mathbf{x}$ is all features and y is a class of a sample. Then the sample is accepted with probability proportional to the misclassification cost of its class. The accepting probability is $\frac{c_{i,j}}{Z}$ where $c_{i,j}$ represents the misclassification cost of classifying a sample from class j predicted as class i, and Z is some constant such that $Z \geq \max[c_{i,j}]$. Under the assumption that the false negative error is more expensive than the false positive error, $Z = \max[c_{i,j}]$ means that all examples from the positive class are kept. After the sampling is done, bagging is performed to improve the results further [170].

Algorithm 3 Costing (S, C) [50]

```
Require: Original dataset $S$,
         Misclassification cost of classes $C = [c_{i,j}]$
  $S' = [\,]$
  for i=1 to t do
    Draw a sample $(\mathbf{x}, y)$ from $S$
    Add $(\mathbf{x}, y)$ to $S'$ with the probability $\frac{c_{y',y}}{Z}$
  end for
  return Sampled set $S'$
```

If a learning algorithm, A, can achieve approximate minimization of classification errors, it is theoretically guaranteed that a cost-minimizing classifier can be created using the learning algorithm, A, and a set, S, using the costing method [50].

4.4 Model Evaluation Criteria

The performance of a binary classifier is typically evaluated by predictive accuracy. With asymmetric misclassification costs, however, the performance of the model should be evaluated based on cost as well as the error rate. The misclassification costs can be expressed as a function of the predicted class and the actual class.

$$cost_{predicted,actual} = f(class_{predicted}, class_{actual})$$

The respective costs according to the difference between $class_{predicted}$ and $class_{actual}$ are represented in the form of a matrix called a cost matrix, as shown in TABLE 4.2. In TABLE 4.2, the diagonals correspond to the accurate predictions where $class_{predicted}$ is identical to $class_{actual}$. Thus, a cost matrix is a square matrix whose diagonals are 0.

Table 4.2: Cost Matrix of Delays Prediction

	Predicted On-time	Predicted Delays
Actual On-time	0	$cost_{delays,on-time}$
Actual Delays	$cost_{on-time,delays}$	0

Then here comes the questions. How do we evaluate the performance of classifiers considering a cost matrix? What is the one representative value that can replace predictive accuracy? The total cost can be defined as a function of the type of errors and the corresponding costs [165].

$$Total\ Cost = f(error, cost_{error})$$

$$= \sum error \cdot cost_{error}$$

$$= cost_{FPR} \cdot FPR + cost_{FNR} \cdot FNR$$

where

$$FPR = \frac{FP}{FP + TN} = \frac{FP}{the\ number\ of\ on\text{-}time\ samples}$$

$$FNR = \frac{FN}{FN + TP} = \frac{FN}{the\ number\ of\ delayed\ samples}$$

$$cost_{FPR} = \frac{cost_{delays,on\text{-}time}}{cost_{on\text{-}time,delays} + cost_{delays,on\text{-}time}}$$

$$cost_{FNR} = \frac{cost_{on\text{-}time,delays}}{cost_{on\text{-}time,delays} + cost_{delays,on\text{-}time}}.$$

In the above equations, TP, TN, FN and TP follow the notation in Table 1.2. The cost of FPR and FNR were taken into account in the previous sections, but the exact value of the costs was not known. Thus, the normalized relative costs are used. The total cost can then be interpreted as a weighted sum of FPR and FNR. $cost_{FPR}$ and $cost_{FNR}$ are the weights that constitute the weighted sum of FPR and FNR. It is known that the false negative error is more expensive than the false positive error. However, the exact amount of costs is not known, so the cost ratios are utilized, and it reflects the relative importance of each class. For example, in the case of the false negative error resulting in four times more misclassification cost than the false positive error, the evaluation function is $0.2 \cdot FPR + 0.8 \cdot FNR$. Setting $w_1 = 0.5$ and $w_2 = 0.5$ gives the same results with the case not considering misclassification cost.

To validate Hypothesis 3, experiment 3 is planned to be executed. The cost-proportionate rejection sampling described in Section 4.3.2 is integrated to create models that take the cost into account. In order to compare and evaluate the costing method, the accuracy and misclassification cost of the machine learning classifiers will be measured with and without utilizing the costing method.

Here are some questions to be answered by experiment 3 in particular.

- Verify the implementation of the costing method with publicly available datasets, for example, KDD-Cup-98
- Performance comparison before and after applying the costing method
- Determine whether the costing method degrades the diagnostic capabilities of the model

A set of validation criteria for experiment 3 is the following.

- Weighted Sum of Errors $Total\ Cost = cost_{FPR} \cdot FPR + cost_{FNR} \cdot FNR$
- Receiver Operating Characteristic (ROC) Curve

Hypothesis 3 will be accepted if the weighted sum of false positive error and false negative error is improved compared to the results when the costing method is not applied.

4.5 Experiments

Figure 4.3 shows the sequence of steps to integrate the sampling algorithm described in Section 4.3 into the classification algorithm described in Section 3.3 and to validate the model. The dataset is generated by merging historical weather data and flight data together. Data generation is followed by the data preprocessing step. Data preprocessing includes normalization and data interpolation. Then the preprocessed dataset is split into training data and test data. The training data is fed into the model after cost-sensitive sampling and used to learn parameters of a predictive function. For prediction, the test set is used to assess the predictive performance and generalization of a classifier. In the end, the model assigns each data point a label and $k-$folds cross-validation is used to detect symptoms of overfitting.

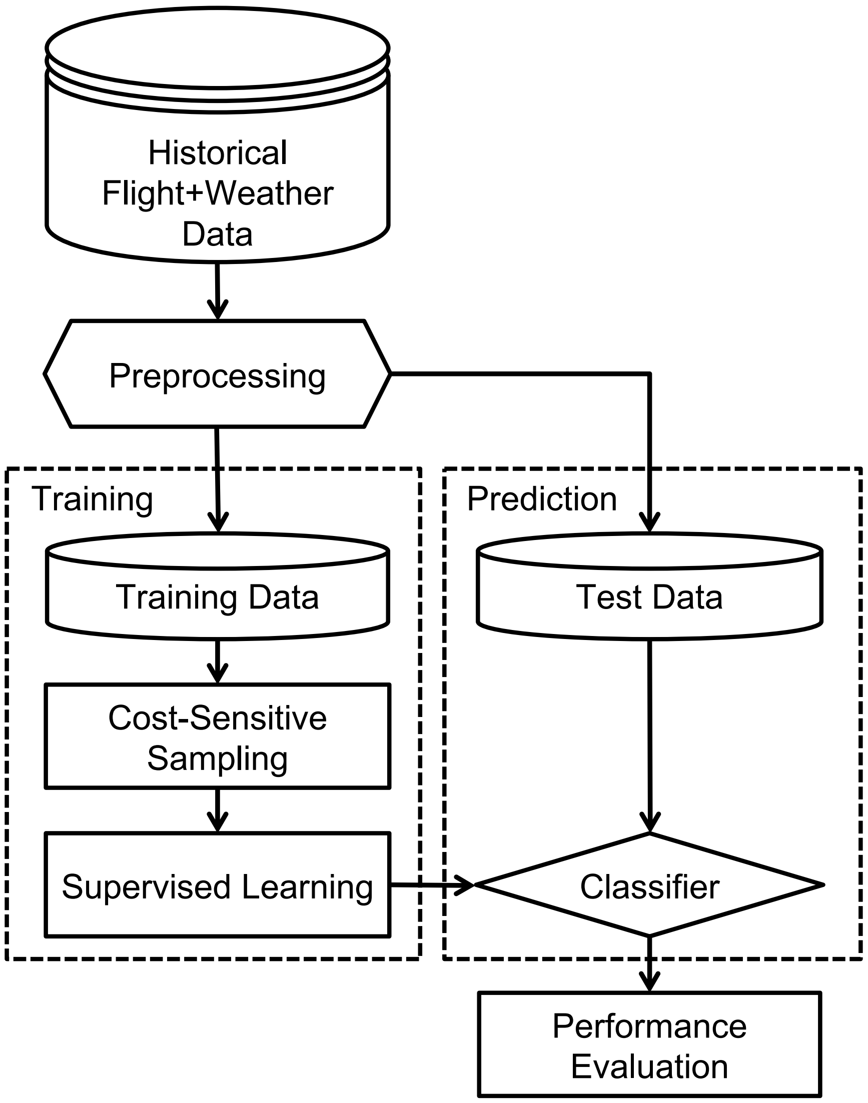


Figure 4.3: Flow chart for Cost-sensitive Individual Flight Delays Prediction

4.5.1 Verification of the Implementation

Before applying the costing method to the air traffic data, it is required to check whether the implementation of the costing is correct. So the publicly available datasets, the KDD-Cup-98 data is used to ensure that the costing method is implemented correctly. The KDD-Cup-98 data is available at the University of California, Irvine KDD Archive [171]. The dataset was provided by the Paralyzed Veterans of America (PVA). The PVA is a nonprofit organization and raises postal funds to provide services to injured American veterans. The PVA sent letters to former donors for fundraising. The amount of donation by former donors in response to the mail is from $\$0$ to $\$200$. On the other hand, the cost of sending a mail to one person is $\$0.68$. In consideration of this, the PVA seeks to maximize the net revenue by predicting who will donate again from former donors, and sending mail selectively.

The KDD-Cup-98 data consists of 95412 samples with 481 fields. The dataset contains information about each previous donor, whether to respond, and the donation amount. If the PVA sends letters to all 95412 people without classification, the net profit is $\$10787.86$. By predicting donors and selectively sending mails only to the actual donor, the PVA can increase the net profits. However, it demands considerable effort because this data is extremely imbalanced. The histogram in Figure 4.4 shows the distribution of the donation amount precisely. The frequencies in each bin are very different, so the y-axis is set as the logarithmic scale. The donation amount is a skewed right distribution with a tail to the right.

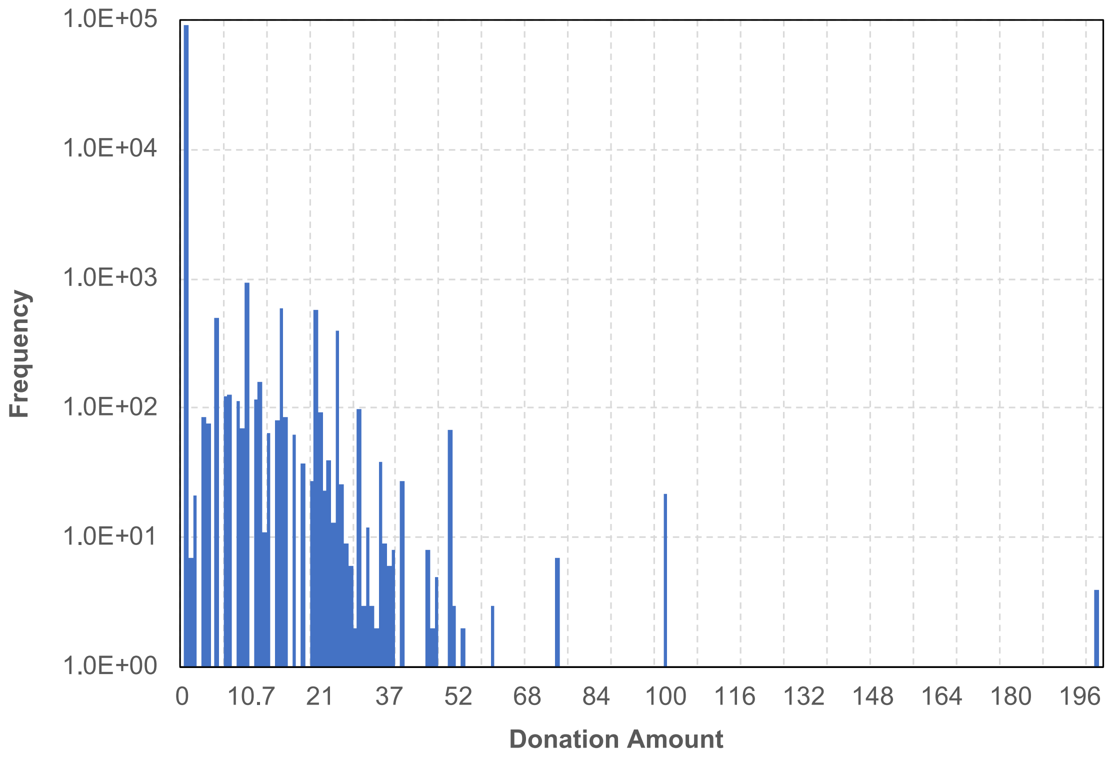


Figure 4.4: Histogram of Donation Amount

Only 5% of the samples are the donors, but the cost of not recognizing the donor is much higher than that of non-donors. Therefore, we can say that it has imbalanced data and asymmetric misclassification cost like the flight delays prediction problem. The misclassification costs of the samples are analyzed in Table 4.3. When a non-donor is predicted as a non-donor, there is no cost at all. If a classifier predicts a non-donor as a donor, there is a loss of $\$0.68$ used to send the letter. Predicting donors is a bit more complicated. Assuming the donation amount of the donor x_i is c_i, the PVA can earn $c_i - 0.68$ if the donor is well predicted. If a donor is predicted as a non-donor, the PVA will not lose anything compared to the state before the prediction. The PVA will miss the donation they could get, but the cost matrix compares the profit or loss before and after the prediction.

Table 4.4 shows the results for multiple classifiers with and without the costing method.

Table 4.3: Cost Matrix for the KDD-98-Cup data

	Predicted Non-donors	Predicted Donors
Actual Non-donors	0	-0.68
Actual Donors	0	Donation Amount $-\ 0.68$

These are the average of 10 repeated experiments. When the costing is not used, the classifiers predicted most of the samples as a non-donor. So, the PVA's profits were very low because the classifiers lost a lot of donors and their donations. On the other hand, the costing method has dramatically improved the results of all classifiers.

As a result of classification using the costing method, the total revenue of PVA increases remarkably. The profit of the PVA improved up to 2 orders of magnitude by utilizing the costing method. It can be seen that the costing method is effective because the average profit of each algorithm is increased by $\$12227$ compared to when the costing was not used.

Table 4.4: Profits with and without Costing on Test Set

	Without Costing ($\$$)	With Costing ($\$$)
Random Forests	104.92	10791.26
AdaBoost	936.08	16104.85
kNN	185.32	10867.46

4.5.2 Threshold Selection

Similar to Section 3.7.2, the misclassification cost for various thresholds was calculated to infer appropriate threshold to distinguish the delayed class from the on-time samples. The weighted sum of FPR and FNR is a measure of the misclassification cost as defined in Section 4.4:

$$Total\ Cost = cost_{FPR} \cdot FPR + cost_{FNR} \cdot FNR$$

where $cost_{FPR} + cost_{FNR} = 1$ and $cost_{FPR}, cost_{FNR} \in [0, 1]$. As stated in Table 1.5, the false negative error can be expressed as a ratio to the false positive error, and the normalized values are used because the exact costs are unknown.

From Reference [51], the theoretical optimal threshold for cost-sensitive learning was derived. According to the research, a cost-sensitive classifier predicts a sample, x as the delayed class when the cost of predicting x as the delayed class is less than that of on-time. This condition can be expressed as follows.

$$\begin{aligned} p(Y = on\text{−}time|x)cost_{delays,on\text{−}time} + p(Y = delays|x)cost_{delays,delays} \\ \leq p(Y = on\text{−}time|x)cost_{on\text{−}time,on\text{−}time} + p(Y = delays|x)cost_{on\text{−}time,delays} \end{aligned}$$

When $p(Y = delays|x)$ is simply refered as τ,

$$\begin{aligned} (1 - \tau)cost_{delays,on\text{−}time} + \tau \cdot cost_{delays,delays} \\ \leq (1 - \tau)cost_{on\text{−}time,on\text{−}time} + \tau \cdot cost_{on\text{−}time,delays}. \end{aligned}$$

The optimal decision rule for the classifier score p and theoretical threshold τ^* is to predict a sample x as delays class when $p > \tau^*$. Rearranging the preceding equation and expressing it for τ^*

$$\tau^* = \frac{cost_{delays,on\text{-}time} - cost_{on\text{-}time,on\text{-}time}}{cost_{delays,on\text{-}time} - cost_{on\text{-}time,on\text{-}time} + cost_{on\text{-}time,delays} - cost_{delays,delays}}$$

where $cost_{ij}$ is the cost of misclassifying samples of class j into class i. Under the assumption that the cost for correct classification is 0, $cost_{on-time,on-time}$ and $cost_{delays,delays}$ are 0, so the theoretical threshold is:

$$\tau^* = \frac{cost_{delays,on\text{-}time}}{cost_{delays,on\text{-}time} + cost_{on\text{-}time,delays}}. \tag{4.1}$$

The misclassification cost of the training samples according to the threshold change can be shown as Figure 4.5. Figure 4.5 is the experimental results when the cost ratio between FPR and FNR is 1:10. According to this, the total misclassification cost is a convex function with a degree of freedom of 1, and there is a unique global minimum. For low threshold values, the cost remained uniformly low, but the cost increased significantly beyond the minimum point.

The key threshold values and the corresponding costs from Figure 4.5 are shown in Table 4.5. As the key threshold values, the theoretical optimal of cost ratio 1:10, the conventional value of, 0.5, and the actual minimum point obtained from the experiment were selected. The theoretical optimal value referred to here is obtained by Equation 4.1. The total cost of the empirical minimum was lower than the cost from the commonly used threshold. The total cost found empirically was even lower than that of the theoretical optimal threshold.

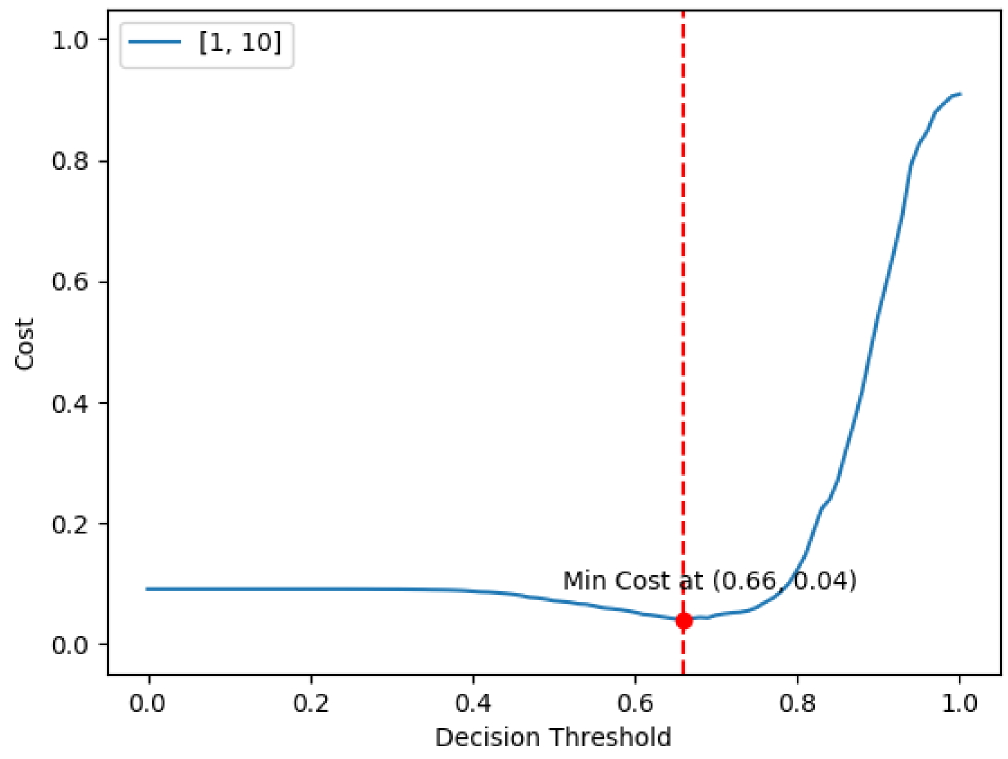


Figure 4.5: Threshold vs. Total Cost of Random Forest with Cost Ratio 1:10

Table 4.5: Key Threshold Values and Total Costs of Random Forest with a cost ratio of 1:10

	Threshold	Total Cost
Conventional	0.5	0.07
Theoretical	0.09	0.09
Empirical Minimum	0.66	0.04

Sensitivity to Cost Ratios

The sensitivity of classifiers is evaluated by varying the cost ratio of FPR and FNR to 1:1, 1:2, 1:5, 1:10, and 1:20. In Figure 3.18, the total misclassification cost according to the threshold is given for each cost ratio. Since the total cost is defined as the weighted sum of FPR and FNR, the value does not exceed 1 for any cost ratio. As the cost of FNR increases, the optimal threshold shifts to the right and the optimal total cost gets lower. Table 4.6 summarizes the important values in Figure 4.6. When the cost of FNR is very high compared to the cost of FPR, the total cost is insensitive to moderately low thresholds. However, if the cost of FPR and FNR are comparable, the threshold should be chosen more carefully to minimize the total cost. This is because when the cost ratio is low, the total cost plot becomes sensitive to small thresholds.

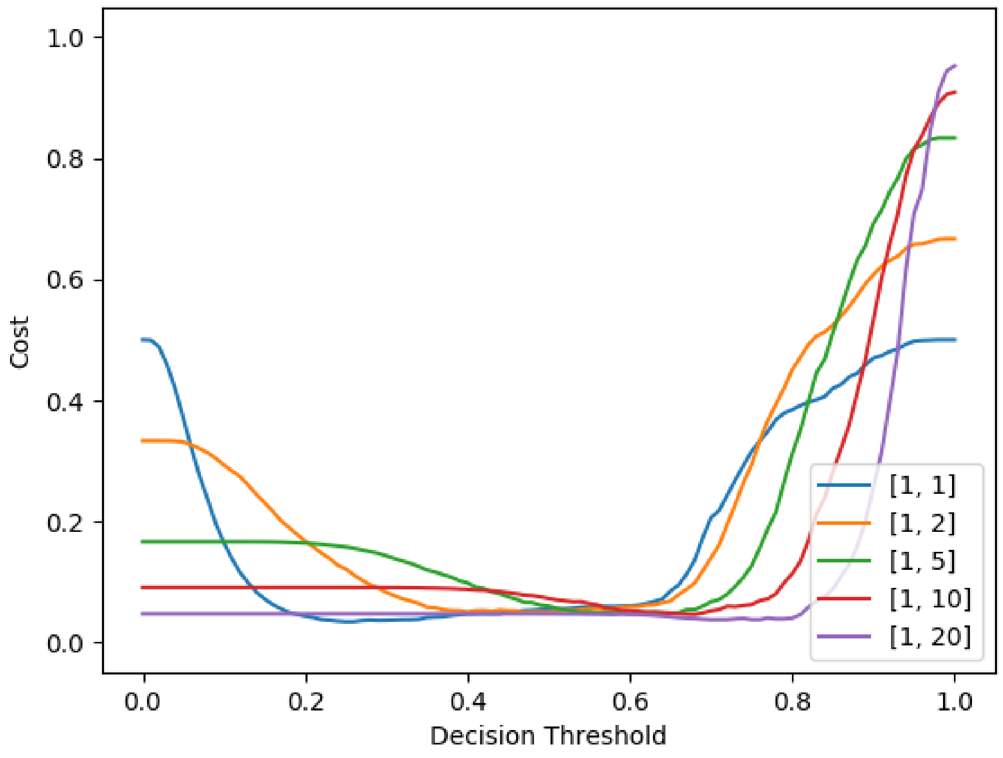


Figure 4.6: Sensitivity of Random Forest to Threshold and Cost Ratio

Table 4.6: Total Cost of Random Forest according to the Threshold for Cost Ratio

Cost Ratio	Threshold		Total Misclassification Cost
1:1	Conventional	0.50	0.05
	Theoretical	0.50	0.05
	Empirical Minimum	0.29	0.03
1:2	Conventional	0.50	0.06
	Theoretical	0.33	0.07
	Empirical Minimum	0.41	0.06
1:5	Conventional	0.50	0.06
	Theoretical	0.17	0.17
	Empirical Minimum	0.60	0.05
1:10	Conventional	0.50	0.07
	Theoretical	0.09	0.09
	Empirical Minimum	0.66	0.04
1:20	Conventional	0.50	0.05
	Theoretical	0.05	0.05
	Empirical Minimum	0.75	0.03

The width of the misclassification cost change according to the threshold becomes larger when the cost ratio is large. Let us look at the trend of cost ratio 1:20 quantitatively. When the threshold is high, the classifier predicts the delayed class fewer, so the FNR increases and the FPR decreases. Since the cost of FNR is 20 times larger than that of FPR, the total misclassification cost increases greatly. If the threshold is small, the FPR is high, and the FNR is low because the positive class is easily predicted. The total cost obtained by weighting FPR and FNR at the ratio $\frac{1}{1+20}$ and $\frac{20}{1+20}$ is very small. Conversely, lower cost ratios result in FPR and FNR almost equally weighted, so the total cost when the threshold is low is relatively large.

4.6 Results

Models that take into account the misclassification costs may be biased towards the more expensive class. The misclassification costs may change the prediction and degrade the accuracy of the model. However, a model that takes into account the misclassification cost can be more practical, even if it is not the most accurate one. Therefore, the performance of models will be evaluated with a focus on the total cost defined in Section 4.4. The total cost provides an integrated understanding of the model's performance from the perspective of FPR, FNR, and costs associated with them. Moreover, the cost ratio shows the relative importance of the delayed class to the on-time class. In other words, the cost ratio is an indicator of how much FNR is relatively more expensive than FPR.

The following experiment is to prove Hypothesis 3 that applying the costing method can reduce the total misclassification cost of delays prediction. Hence, the validity of Hypothesis 3 can be tested by comparing the total cost of the models with the costing method to the total cost of the models that do not use it. Figure 4.7 - Figure 4.9 plot the total cost of each model for the test data, which varies with the cost ratio of FNR to FPR. The cost ratio between FNR and FPR is increased from 1 to 10. It means false negative is 1 to 10 times more expensive than false positive.

For all models without the costing method, the total cost increased as the cost ratio increased. When the costing is not applied, the models are biased towards the on-time class because the on-time class is the majority of data. Therefore, the models have high FNR and a low ability to find the delayed class. The value of FNR was relatively large compared to FPR, and FNR increased with increasing cost ratio. As a result, the models showed a high total cost for the high cost ratio. This result is caused by training the models with the data that does not take into account the cost differences of the classes. This is a common phenomenon in all models, but the total cost of kNN is particularly small compared to the other two.

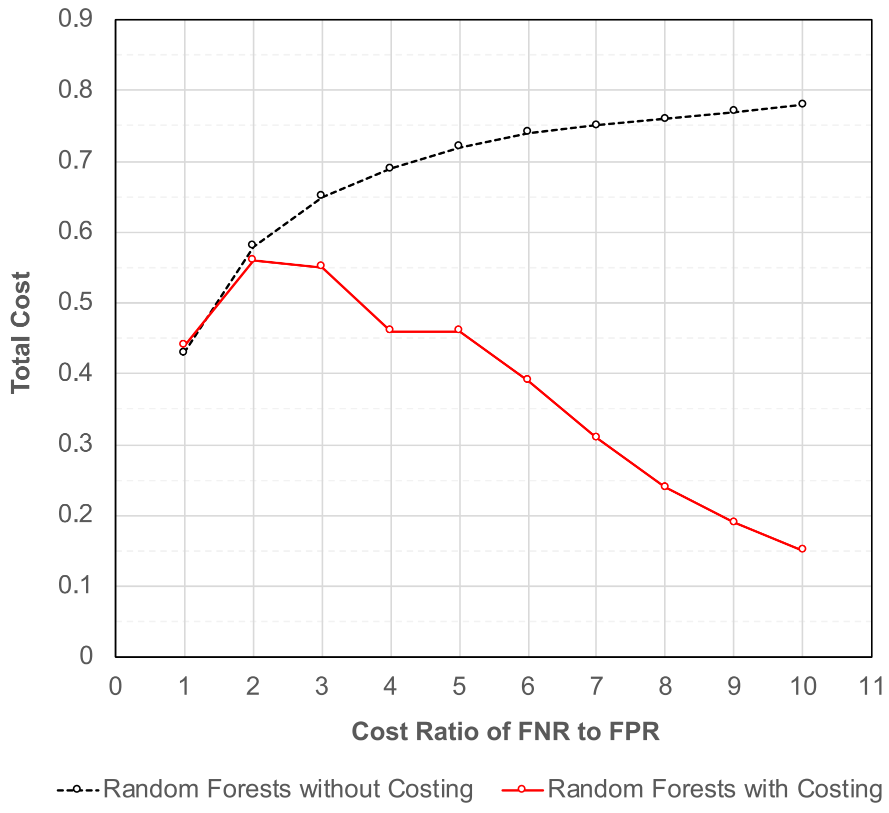


Figure 4.7: Total Cost of Random Forests according to the Costing Method Application

On the other hand, the total cost of models has a different tendency when the costing is applied. For all algorithms, the cost ratio $1:2$ made the total cost increase. As the ratio changed from $1:1$ to $1:2$, the number of delayed samples in the subset increased. As a result, the model became more sensitive to the delayed class, but the decrease in FNR was not significant compared with the increased FPR. It caused the total cost, $0.33 \cdot FPR + 0.66 \cdot FNR$, increased. At a cost ratio of $1:3$ or higher, the total cost begins to decrease sharply. The cost ratio adjusted the distribution of the number of samples per class in the subset to change the weight between the on-time and the delays class. The high cost ratio gives extra weight to the samples of the delayed class when training the classifier.

The critical point in Figure 4.7 - Figure 4.9 is the difference in the total cost with or without sampling. The higher the cost ratio, the greater the difference in total cost

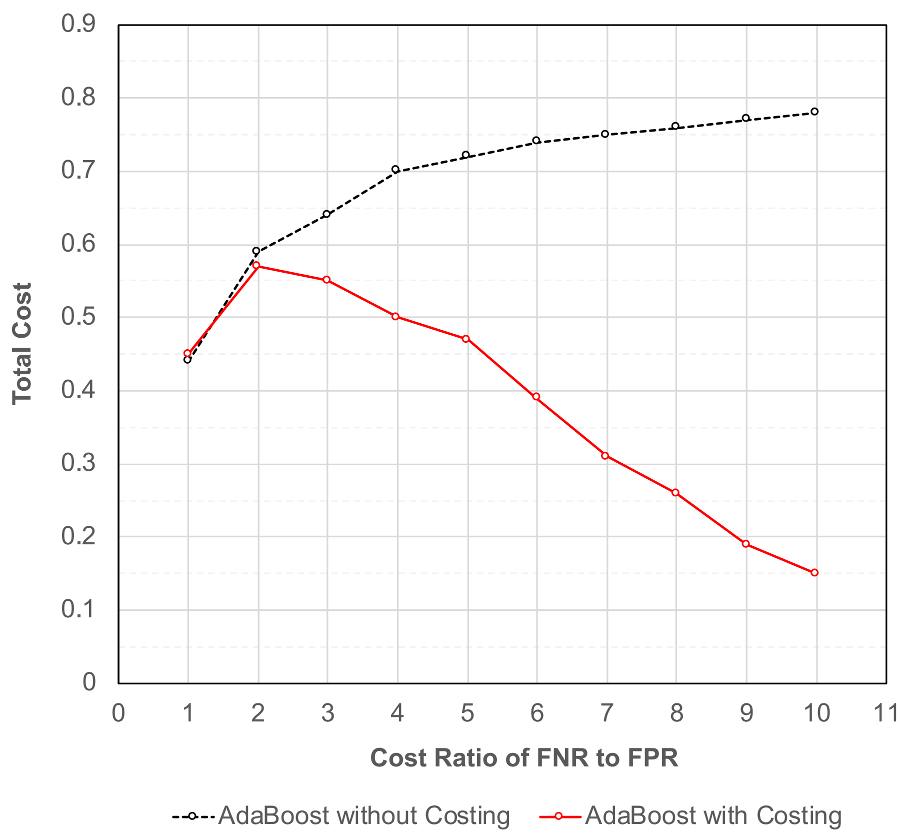


Figure 4.8: Total Cost of AdaBoost according to the Costing Method Application

depending on whether or not the costing is applied. The costing is effective because the total cost between models with and without the costing method is significantly different. Therefore, it is advisable to actively consider the use of the costing method when the cost of FNR is much larger than FPR.

To quantitatively analyze the effect of adjusting the distribution by the costing, FPR, FNR, and the total cost for Random Forests are plotted in Figure 4.10. At a typical threshold range of 0.4 to 0.6, FNR dropped significantly as the cost ratio increased from 1: 1 to 1: 10. Although there was an adverse effect of FPR increment, the total cost was reduced because the weight of the FPR was small. The result is consistent with the analysis of Figure 4.7 - Figure 4.9.

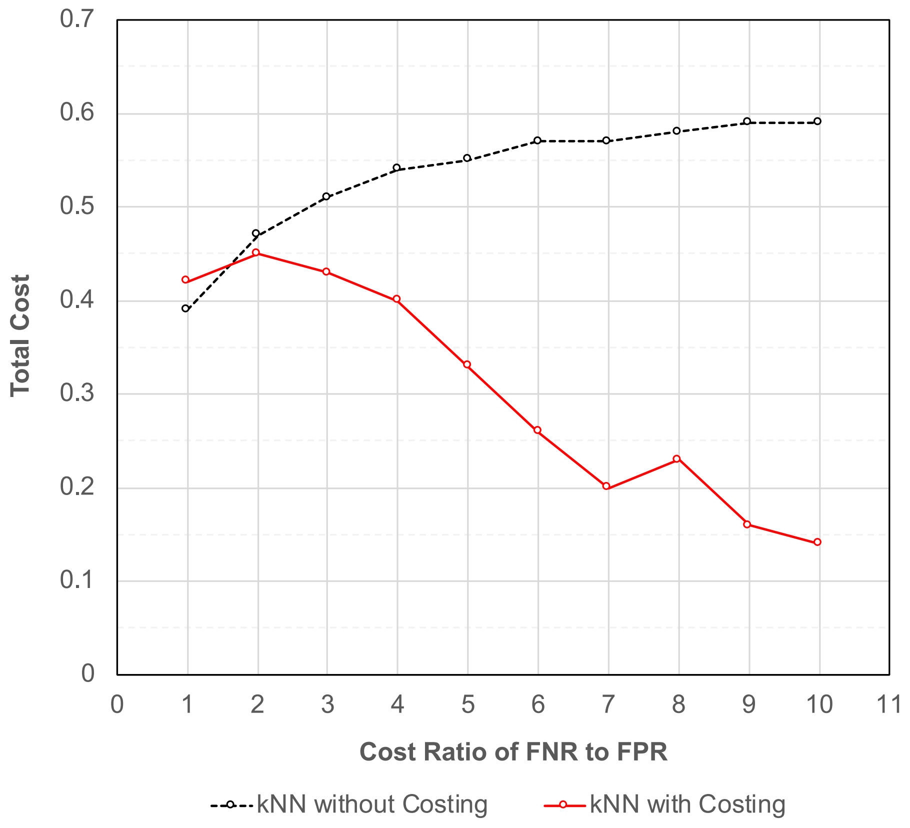
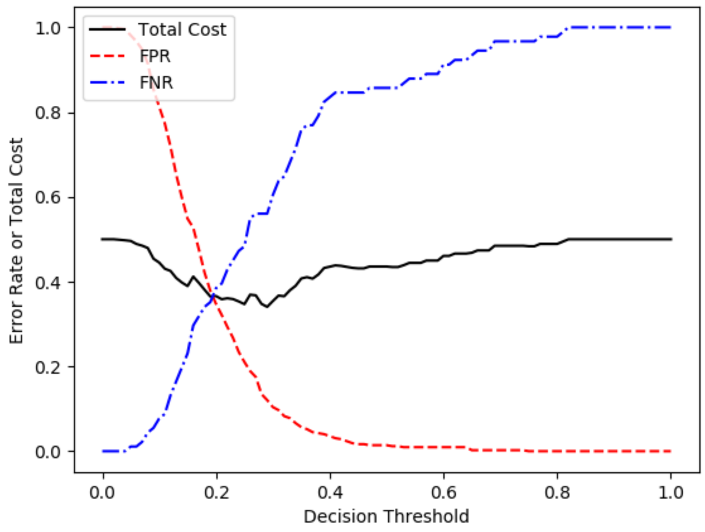
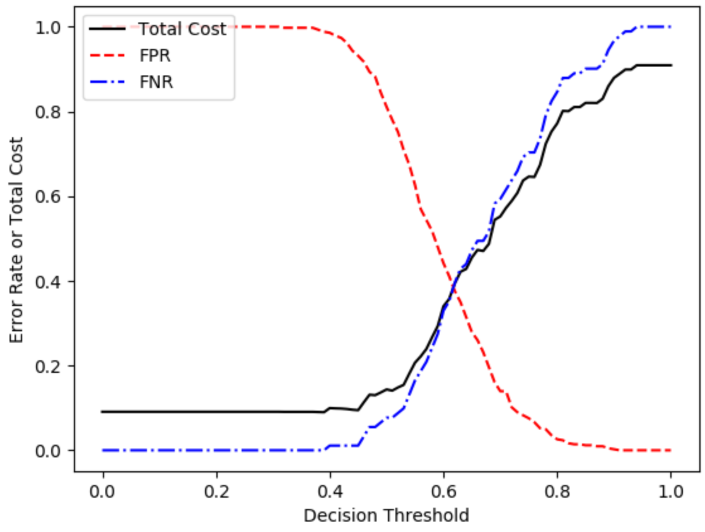


Figure 4.9: Total Cost of kNN according to the Costing Method Application



(a) 1:1

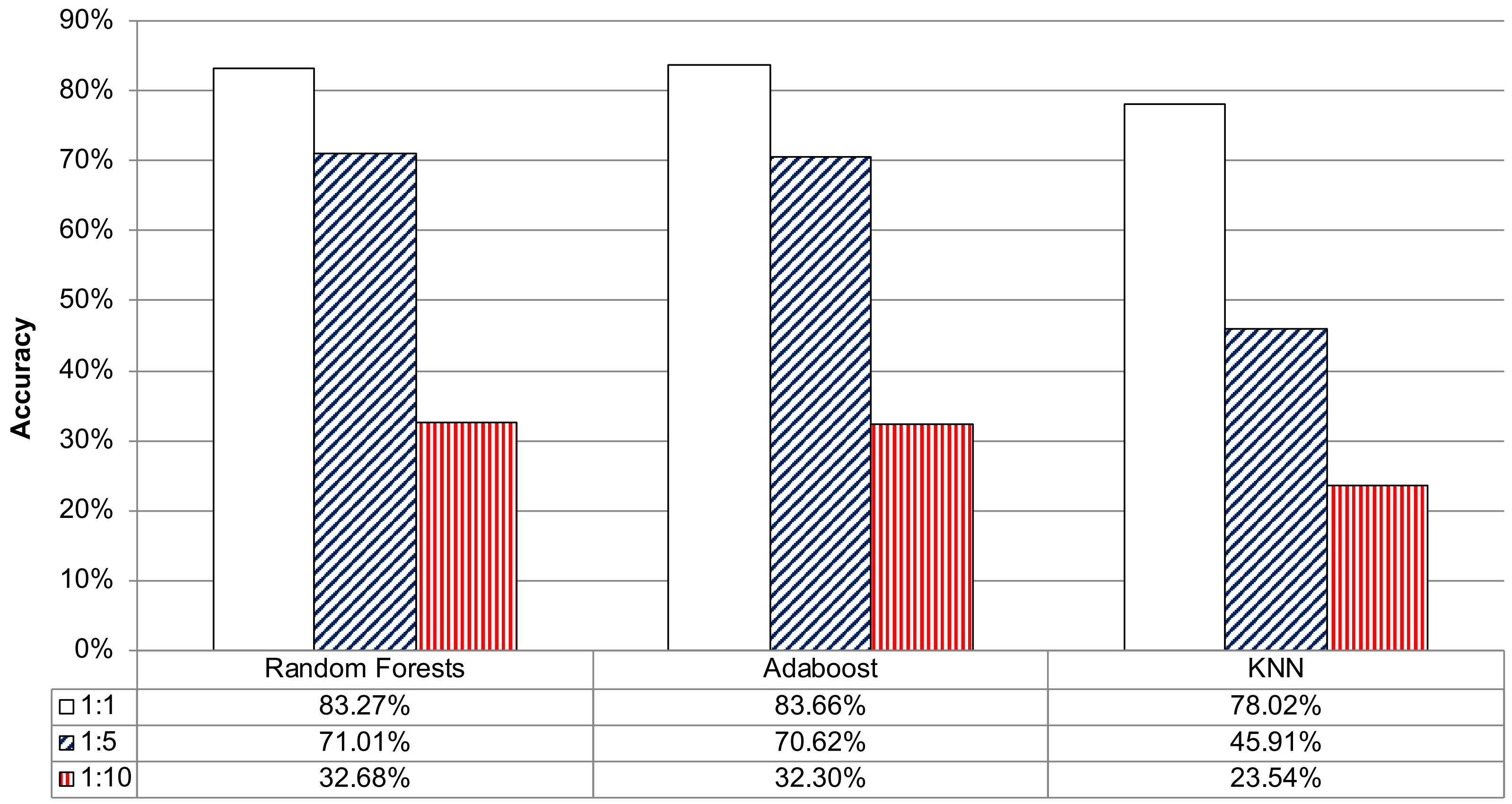


(b) 1:10

Figure 4.10: Total Cost, FPR and FNR of Random Forest according to the Threshold Value when the Cost Ratio is 1:1 and 1:10

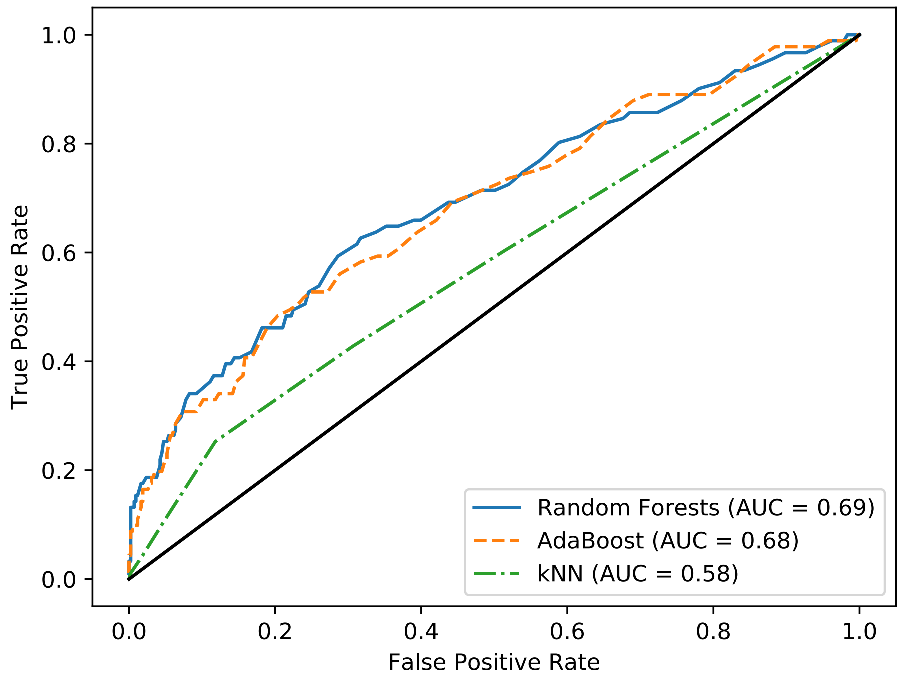
The prediction accuracy of the machine learning algorithms for various cost ratios is shown in Figure 4.11. When the cost ratio is $1:1$, the classifiers could have a decent accuracy because they did not take the misclassification cost into account. After applying the costing method, the accuracy of all algorithms was reduced although there was a difference in variability. The reason can be found in the imbalanced dataset. The delayed class is the minority class taking only $20-25\%$ of the entire dataset, but the misclassification cost is much higher than the on-time class. Models generated with particular attention to the minority class are less capable of classifying the majority class than models that do not. As a result, the number of correctly classified samples decreased, and accuracy decreased. We sacrificed accuracy and incorporated the costing method to bias the models toward a more expensive class deliberately. In this process, we could use the cost ratio to control the degree of attention in the minority class. As in Figure 4.11, the higher the cost ratio, the lower the accuracy.

The Receiver Operating Characteristic (ROC) curves in Figure 4.12 has been plotted with varying cost ratio between FPR and FNR. The ROC curve can be used to determine if the costing method is detrimental to the performance of the binary classifiers. Figure 4.12(a) corresponds to the case where the costs for FPR and FNR are the same. This means that the training data itself has trained the model without any sampling technique. Figure 4.12(b) and Figure 4.12(c) represent the ROC curve for the cost ratio of 1:5 and 1:10, respectively. The performance of the model under various cost ratios from 1:1 to 1:10 was observed, and no significant performance degradation was found in the ROC curve. The shape of the ROC curves in Figure 4.12(b) and Figure 4.12(c) has been changed compared to Figure 4.12(a), but the area under the curve (AUC) remained almost the same. This means that asymmetric mislabeling costs can be taken into consideration without losing the diagnostic ability of binary classifiers.



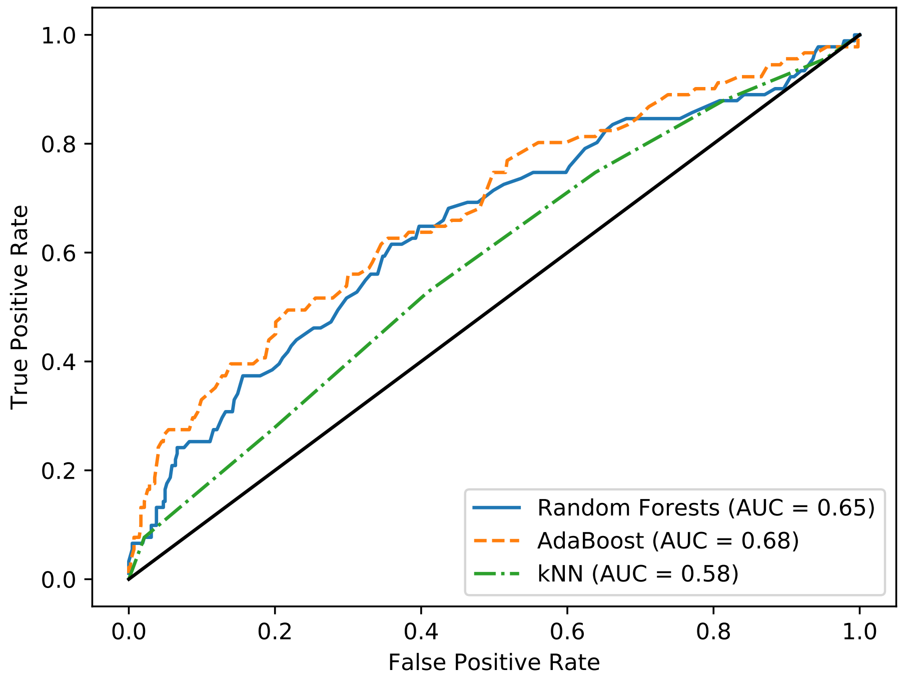
	Random Forests	Adaboost	KNN
□ 1:1	83.27%	83.66%	78.02%
▨ 1:5	71.01%	70.62%	45.91%
🟥 1:10	32.68%	32.30%	23.54%

Figure 4.11: Accuracy of Models according to Various Cost Ratios when Threshold = 0.5



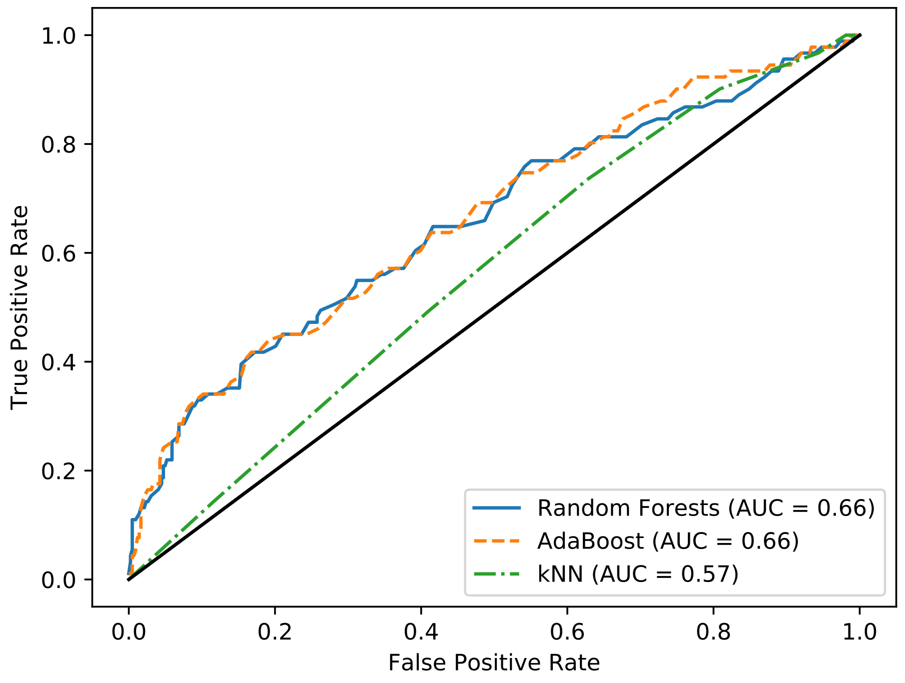
1.0
0.8
0.6
0.4
0.2
0.0
True Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate
Random Forests (AUC = 0.69)
AdaBoost (AUC = 0.68)
kNN (AUC = 0.58)

(a) 1:1



1.0
0.8
0.6
0.4
0.2
0.0
True Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate
Random Forests (AUC = 0.65)
AdaBoost (AUC = 0.68)
kNN (AUC = 0.58)

(b) 1:5



(c) 1:10

Figure 4.12: ROC with different cost ratio

4.7 Chapter Summary

This chapter proposed the cost-sensitive binary classifier to identify individual flight delays. It was to prove Hypothesis 3, which stated that the predictive model with the cost-proportionate rejection sampling (costing) method can find the optimal prediction that minimizes the total cost of misclassification. The costing method manipulates the distribution of samples in the dataset according to the misclassification cost. Next, the costing method was integrated into general machine learning algorithms. In this way, general machine learning algorithms could be transformed into the cost-sensitive classifiers and the costs of misclassification could be taken into account.

As a result of the misclassification cost analysis of the on-time class and the delayed class, the relative ratio between the two was used because their exact values were unknown. Performance changes in the model due to the misclassification costs were measured for the various cost ratios. As a representative metric for the performance of the cost-sensitive classifier, a weighted sum of false positive error rate and false negative error rate was utilized. An in-depth analysis of the metrics and ROC curves showed that the cost-sensitive classifiers were able to account for asymmetric misclassification costs without losing their diagnostic functionality as binary classifiers.

CHAPTER 5
CONCLUSION

This chapter concludes the dissertation. Section 5.1 summarizes the contributions of this dissertation, and Section 5.3 provides suggestions for future extension of the study.

5.1 Summary and Contribution

The main purpose of this paper is to develop a multi-level methodology that can reduce delays in the terminal area through the management of air traffic flow. From the general literature review of air traffic management, we chose an approach that looked at the airport terminal area from macroscopic and microscopic viewpoints. Aiming at an efficient and effective air transportation system, the main research objective of this dissertation is defined.

- Research Objective: To develop a methodology that can efficiently and practically predict macro and micro-level air traffic flow in the terminal area

As the reason for the delays in the macroscopic perspective is the imbalance between airport capacity and demand, the better way to predict airport capacity was explored. Delays from the microscopic point of view are caused by issues affecting the operation of individual flights, such as weather conditions or mechanical defects. In this context, three research questions are then specified, and corresponding hypotheses are formulated based on the gaps in existing methods. The overall research process is summarized in Figure 5.1.

At the macroscopic level, Research Question 1 has been asked about how to improve the prediction performance of the airport's hourly capacity.

- Research Question 1: What is an accurate and computationally efficient method to predict the hourly capacity of an airport?

Through reviewing previous papers, we found that an artificial neural networks approach does not require labor-intensive setup or mathematical assumptions unlike analytical models or simulation models and it is excellent in accuracy. For these reasons, the artificial neural networks algorithms were introduced to analyze the hourly capacity of the airport. In this regard, Hypothesis 1 has been established, and the experiments were conducted to verify Hypothesis 1.

- Hypothesis 1: If suitable artificial neural networks are constructed using historical weather and airport traffic capacity data, then one could accurately predict the hourly capacity for a given airport under different scenarios.

Multi-Layer Perceptron, Recurrent Neural Networks, and Long Short-Term Memory trained with Hartsfield-Jackson Atlanta airport data between 2013 and 2017 showed good accuracy. To test Hypothesis 1, we checked the training and testing errors of these models. The proposed artificial neural networks approach for the macroscopic level modeling provided an accurate and computationally efficient way to predict the hourly capacity of an airport. Furthermore, we have discovered the generalization capability of the artificial neural networks models. The models trained with the Hartsfield-Jackson Atlanta airport data were able to predict the capacity of the Boston Logan International Airport (BOS) well. It was a practical approach in that one model could be re-used at another airport instead of building each airport-specific model separately.

The analysis of the air traffic flow in the terminal area from a microscopic point is about the prediction of individual flights delays. This led to Research Question 2.

- Research Question 2: What is an appropriate method to accurately predict the on-time performance of aircraft?

Research Question 2 is to improve the flight delays prediction. Machine learning algorithms have been proposed as an advanced approach to classifying the delays of individual flights. In the flight delays prediction problem, the imbalance in the number of samples between the delayed class and the on-time class is an issue. If machine learning algorithms are trained with a dataset whose distribution between classes is adjusted by an appropriate sampling method, they are good at handling the imbalance. From this, Hypothesis 2 was formulated as follows.

- Hypothesis 2: Flight delays prediction using the ensemble methods of machine learning provides interpretability and imbalance data handling, but accuracy is as good as the existing methods.

Models using Random Forest, AdaBoost, and k-Nearest Neighbors are proposed as machine learning algorithms for classifying delays in individual flights. Also, Synthetic Minority Over-sampling TEchnique (SMOTE) is proposed as a sampling method for imbalanced data and was integrated into machine learning models. To verify Hypothesis 2, the performance of the proposed models was evaluated in various aspects, such as accuracy and the Receiver Operating Characteristic (ROC) curve. The models were able to distinguish the samples of the delayed class well from the samples of the on-time class. In addition, visualization of the proposed models' decision rule provided insight to interpret the prediction results to the user.

Research Question 3 concerns the expansion of the individual flight delay prediction model discussed in Research Question 2. Research Question 3 is to consider and minimize the misclassification cost of the delays prediction problem.

- Research Question 3: When forecasting on-time performance of flights, what is an appropriate method to handle imbalanced misclassification costs and minimize the total cost from incorrect prediction?

Research Question 3 arises from the fact that the loss that occurs when a sample of the delayed class is misclassified as the on-time class is significantly higher than the loss of opposite case. The practical delays prediction model should be able to make predictions that optimize the total misclassification cost. The cost-proportionate rejection sampling method has been proposed as a solution to this problem. Unlike conventional classifiers, the cost-proportionate rejection sampling method allows taking into account asymmetric misclassification cost when making a decision.

- Hypothesis 3: If the cost-proportionate rejection sampling method is applied to the predictive model, then the model can find the optimum prediction with the lowest total cost due to misclassification.

The cost-proportionate rejection sampling method has been applied to individual delays prediction models to demonstrate its effectiveness. By using the cost-proportionate rejection sampling method, we were able to achieve good predictive performance and reduce the total costs due to misclassification. Also, the asymmetric misclassification costs were reflected in the performance evaluation. In this way, it became possible to evaluate the practical performance beyond just assessing the accuracy of the model.

A methodology was developed to predict macro and micro air traffic flow in the terminal area through research questions, hypotheses, and verification. The whole process of the methodology is shown in Figure 5.2. The left and right boxes are the capacity prediction model and the delays classification model, respectively. A detailed description of each model is given in Section 2, Section 3, and Section 4. Each model may be used separately, or both models may be combined into one integrated capability. The part marked in red in the figure is a bridge between the macro and the micro model. Though it is not addressed in the current study, future studies can use the macro model to help the micro model improve accuracy. This can be done by adding a capacity level to the data feature for delays prediction. Then the delays prediction model is trained with the data including the capacity level. The test data of the delay prediction model includes the capacity values derived from the capacity prediction model. By doing so, the congestion level at the origin and destination airports can be considered when predicting the delays of a particular flight. As a result, the performance of the model will improve as it will be able to predict not only the delays caused by the weather but also delays caused by airport congestion.

The methodology proposed in this study is expected to contribute to the efficient operations in air traffic management in the NAS. Through the method proposed, this study presented a way to make good use of the air traffic data accumulated over decades. The data can be transformed into appropriate information for various stakeholders. An accurate prediction of the airport capacity and individual delays will assist stakeholders in taking more efficient decisions. The scope of the methodology is the air traffic flow of the national airspace system. However, its application will not be limited to the air traffic flow in the United States if appropriate data is available from other countries.

5.2 Limitations

The accuracy of the methods developed in this dissertation is limited by the data used in model training. Key aspects of the data that affect the accuracy of the models are followings.

- Quality of Data: The performance of the models is directly influenced by the quality of the data. The number of data samples available for training is limited since the hourly airport capacity and the number of flights are physically limited. Also, the number of data fields for each flight was limited in this research because the public data was used. If the detailed data for each flight step other than the scheduled departure/arrival time is available for training, the predictive performance of the model is expected to improve. By doing so, it will be easier to estimate the total arrival delay by estimating the delay that occurs at each stage.

- Bounds of Weather Data: Convective weather fluctuates due to its stochastic nature. The model is trained with data for a specific period, and the performance of the model is limited to the bounds of the weather data for that period. The model is accurate if the weather forecasts for capacity prediction or delays classification are within the historical weather bounds. However, if the weather forecast exceeds the bounds of historical weather data, the application of the model is limited. For example, when there is a record-breaking heavy rainfall or heavy snowfall, the predictive performance of the model can be significantly reduced.

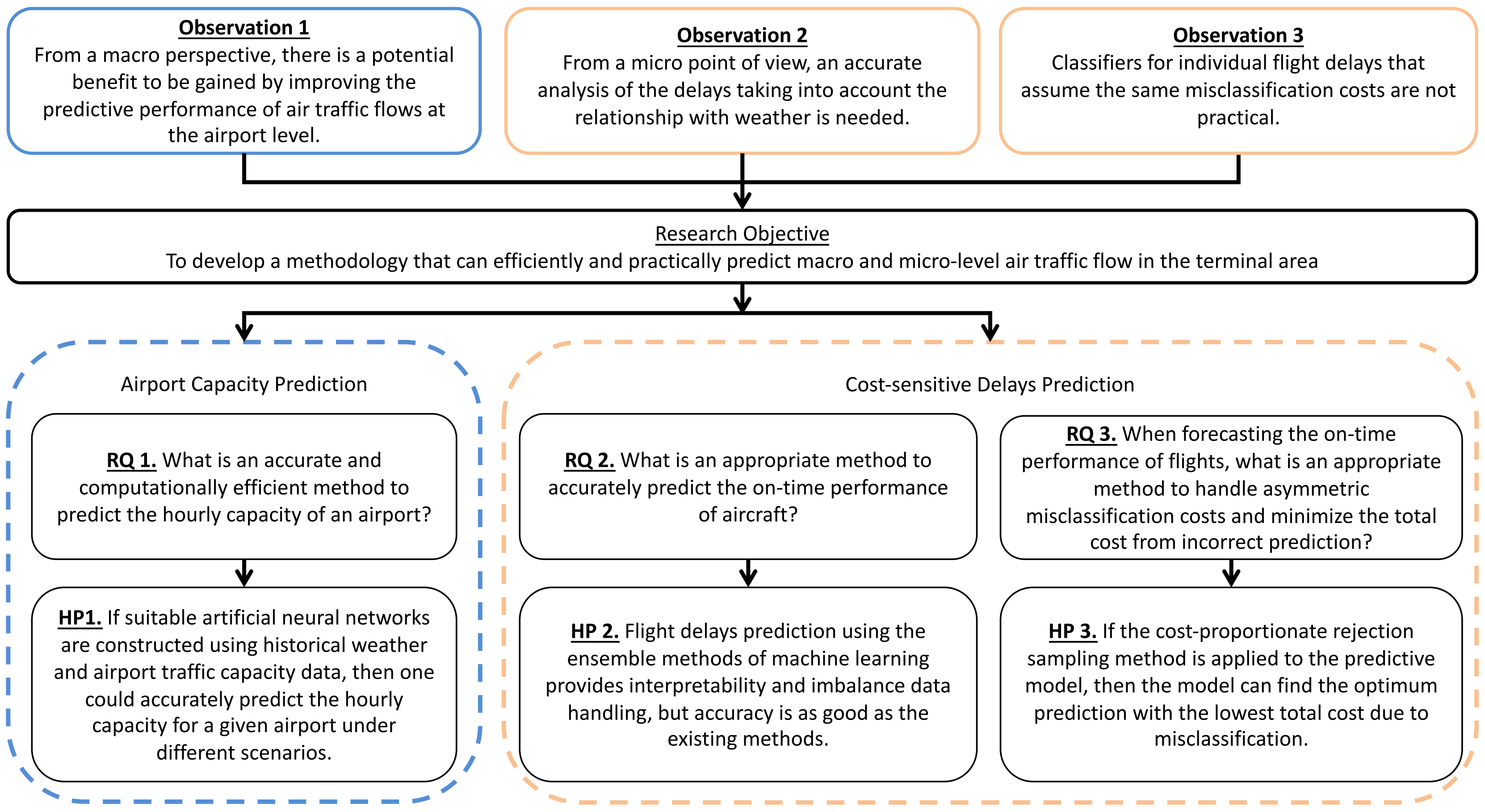

Observation 1
From a macro perspective, there is a potential benefit to be gained by improving the predictive performance of air traffic flows at the airport level.
Observation 2
From a micro point of view, an accurate analysis of the delays taking into account the relationship with weather is needed.
Observation 3
Classifiers for individual flight delays that assume the same misclassification costs are not practical.
Research Objective
To develop a methodology that can efficiently and practically predict macro and micro-level air traffic flow in the terminal area
Airport Capacity Prediction
RQ 1. What is an accurate and computationally efficient method to predict the hourly capacity of an airport?
HP1. If suitable artificial neural networks are constructed using historical weather and airport traffic capacity data, then one could accurately predict the hourly capacity for a given airport under different scenarios.
Cost-sensitive Delays Prediction
RQ 2. What is an appropriate method to accurately predict the on-time performance of aircraft?
HP 2. Flight delays prediction using the ensemble methods of machine learning provides interpretability and imbalance data handling, but accuracy is as good as the existing methods.
RQ 3. When forecasting the on-time performance of flights, what is an appropriate method to handle asymmetric misclassification costs and minimize the total cost from incorrect prediction?
HP 3. If the cost-proportionate rejection sampling method is applied to the predictive model, then the model can find the optimum prediction with the lowest total cost due to misclassification.

Figure 5.1: Summary of the Research Process

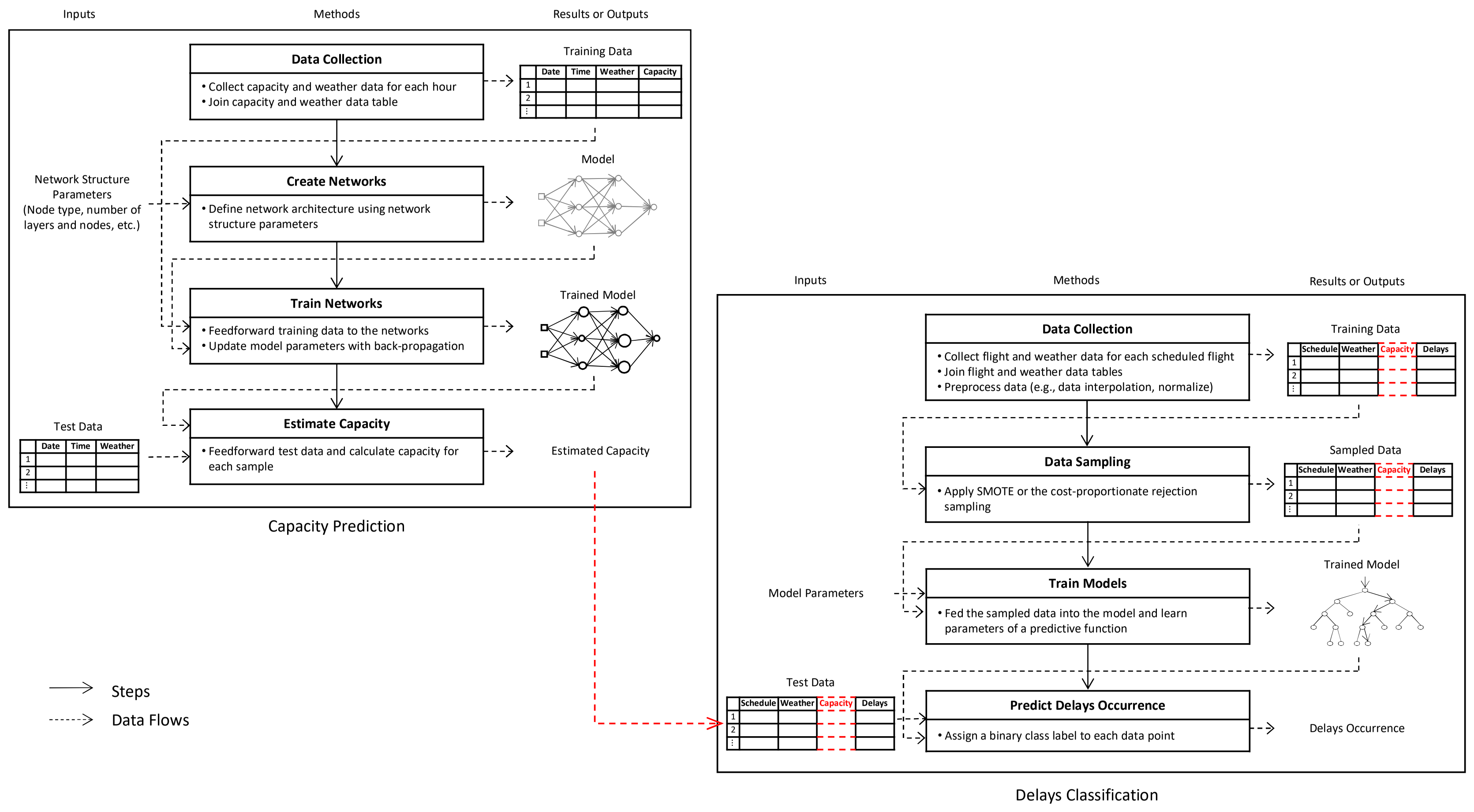

Inputs
Methods
Results or Outputs
Data Collection
• Collect capacity and weather data for each hour
• Join capacity and weather data table
Training Data
Date
Time
Weather
Capacity
Network Structure Parameters (Node type, number of layers and nodes, etc.)
Create Networks
• Define network architecture using network structure parameters
Model
Train Networks
• Feedforward training data to the networks
• Update model parameters with back-propagation
Trained Model
Test Data
Estimate Capacity
• Feedforward test data and calculate capacity for each sample
Estimated Capacity
Capacity Prediction
Steps
Data Flows
Inputs
Methods
Results or Outputs
Data Collection
• Collect flight and weather data for each scheduled flight
• Join flight and weather data tables
• Preprocess data (e.g., data interpolation, normalize)
Training Data
Schedule
Weather
Capacity
Delays
Data Sampling
• Apply SMOTE or the cost-proportionate rejection sampling
Sampled Data
Model Parameters
Train Models
• Fed the sampled data into the model and learn parameters of a predictive function
Trained Model
Test Data
Predict Delays Occurrence
• Assign a binary class label to each data point
Delays Occurrence
Delays Classification

Figure 5.2: Steps and Data Flows of the Proposed Methodology

5.3 Recommendations for Future Work

There is still room for improvement in system modeling and experiment. The following items are recommended for future research.

Future work on airport capacity prediction includes creating an artificial neural network model that can integrate weather forecasts uncertainty explicitly. It is known that airport capacity is greatly affected by terminal weather. Although deterministic weather has been used in this study, there is always uncertainty in the weather forecasts needed to predict future capacity. If the performance of the model changes due to the uncertainty of the weather forecast, exploring it will help improve the robustness of the model.

From the perspective of the generalizability of the capacity prediction model, it can be one of the future works to create a universal model that can be applied to all airports in the National Airspace System. Through this study, we have discovered the generalization capability of the artificial neural networks model. Therefore, the next step would be to find a systematic way to scale the model to the NAS.

In addition, a study on the cost-sensitive binary classification can be extended to cost-sensitive multiclass classification or regression. It would be a much better model if we could estimate the delay minutes, considering the costs that can occur depending on the degree to which the predicted value differs from the actual value. For example, the cost of incorrect regression may be quantified as follows:

$$cost = \begin{cases} p(actual - predicted) & if(actual \geq predicted) \\ q(predicted - actual) & otherwise \end{cases} \tag{5.1}$$

where p and q are positive constants. The larger the difference between the actual arrival time and the predicted arrival time of the model, the greater the loss. We can consider the approach of giving a high weight when the estimated time minus the actual arrival time is substantial.

Quantification of the cost of misclassification can also be a research area. In this paper, we used the ratio between the delayed class and the on-time class without an estimate of the magnitude of the loss due to misclassification. If we can set the scope of the study and estimate the actual loss that can occur within that scope, we can make a more accurate model.

Appendices

APPENDIX A
FLIGHT OPERATIONS CONTROL

A.1 14 CFR Part 121 Operating Requirements: Domestic, Flag, and Supplemental Operations

Sec. 121.533 Responsibility for operational control: Domestic operations.

(a) Each certificate holder conducting domestic operations is responsible for operational control.

(b) The pilot in command and the aircraft dispatcher are jointly responsible for the preflight planning, delay, and dispatch release of a flight in compliance with this chapter and operations specifications.

(c) The aircraft dispatcher is responsible for

(1) Monitoring the progress of each flight;

(2) Issuing necessary information for the safety of the flight; and

(3) Cancelling or redispatching a flight if, in his opinion or the opinion of the pilot in command, the flight cannot operate or continue to operate safely as planned or released.

(d) Each pilot in command of an aircraft is, during flight time, in command of the aircraft and crew and is responsible for the safety of the passengers, crewmembers, cargo, and airplane.

(e) Each pilot in command has full control and authority in the operation of the aircraft, without limitation, over other crewmembers and their duties during flight time, whether or not he holds valid certificates authorizing him to perform the duties of those crewmembers.

A.2 Planning and Operation Management of Flight by Stage

A.2.1 Preflight Planning

- D-6 months to 1 month [157]:

  1) To allow passenger reservations, the Passenger Schedule is generated. The Passenger Schedule includes origin airport, destination airport, any alternate airports, departure time, arrival time and aircraft type.

  2) To support the Passenger Schedule, the Crew Trips Schedule and the Aircraft Rotations Schedule are determined. These two schedules describe the activities of aircraft and crew members, the major resources of the airline.

- D-1 month to 1 week [157]: The operations manager of each airline creates to handle the resource of the individual stations along with the generated passenger schedule, the crew trips schedule and the aircraft rotations schedule. The resources for each station to be scheduled include gates and all ground operating equipment. In general, the schedules for the ground operating equipment are updated as needed.

- D-3 days to 1 day [172] [173]: Given the fleet of aircraft and the flights to be operated, aircraft routings are determined by scheduling the aircraft on the schedule and maintenance lines so that all aircraft are properly maintained.

- D-6 hours to 4 hours [172] [157]:

  1) The optimal routing options are determined.

  2) Load information collection and the preliminary weight and balance calculations are initiated.

- D-4 hours to 90 minutes [172] [157]: The final routing is determined after considering the followings. Dispatchers should monitor the progress of the following items for the flight. If dispatchers recognize that the scheduled activity for the following items is deviated from the plan and is operating abnormally, they will notify the Airline Operations Controller (AOC) of this information. Then the AOC can exchange resources, delay the flight, or replace the delayed aircraft with spare resources.

1) Dispatchers ensure that aircraft authorized by the regulations is assigned to the flight.
2) Aircraft maintenance status and requirements
3) Enroute winds and weather conditions (turbulence, convection, etc.)
4) ATC costs
5) Fuel requirements calculation
6) Dispatchers monitor the cockpit and flight crew hours to ensure compliance with regulations such as FAR 121.471.
7) Notices to Airmen (NOTAMs), indicating the real-time and abnormal status of the NAS components [174]
8) Aircraft minimum equipment list

- D-75 minutes to 30 minutes [172] [157]:

  1) After all of the previous items have been met, a dispatch release is issued under CFR 121.687.
  2) The flight plan is submitted to the FAA and the Air Navigation Service Provider (ANSP).
  3) The pilot in command must accept the flight plan. Without an acceptance of the pilot in command, the flight cannot proceed.

A.2.2 Enroute Monitoring

After submitting the flight plan, the pilot in command and dispatchers are responsible for flight monitoring.

- During the flight, the pilot in command and dispatchers maintain rapid and reliable communication to ensure the safe flight.
- In particular, dispatchers have a task of keeping the flight up-to-date with the current en-route conditions, the destination airport and the alternate airport condition.
- In the event of an unexpected contingency such as weather deterioration during flight, dispatchers become the primary coordinator and search for the appropriate response [172]. Events that can occur during a flight and affect flight safety are not limited to weather changes but include such things as mechanical defects and ATC reroutes.
- When the irregular operation happens, dispatchers and pilot in command should revise their original plan and present a new plan to avoid disruptions.

APPENDIX B
NATIONAL AIRSPACE SYSTEM

B.1 Visual Meteorological Conditions [175]

To be classified as visual meteorological conditions (VMC), the visibility and the minimum distance from the cloud bottom must be met. When the minimum distance is not met, it is instrumental meteorological conditions (IMC) and should follow instrument flight rules (IFR). The minimum requirements are shown in the following table.

Table B.1: VMC minima

Altitude band	Airspace class	Flight visibility	Distance from cloud
At and above 3050 m (10000 ft) AMSL	A B C D E F G	8 km	1500 m horizontally, 300 m (1000 ft) vertically
Below 3050 m (10000 ft) AMSL and above 900 m (3000 ft) AMSL, or above 300 m (1000 ft) above terrain, whichever is the higher	A B C D E F G	5 km	1500 m horizontally, 300 m (1000 ft) vertically
At and below 900 m (3000 ft) AMSL, or 300 m (1000 ft) above terrain, whichever is the higher	A B C D E	5 km	1500 m horizontally, 300 m (1000 ft) vertically
	F G	5 km	Clear of cloud and with the surface in sight

B.2 Visual Flight Rules [175]

1. Except when operating as a special VFR flight, VFR flights shall be conducted so that the aircraft is flown in conditions of visibility and distance from clouds equal to or greater than those specified in Table B.1.

2. Except when a clearance is obtained from an air traffic control unit, VFR flights shall not take off or land at an aerodrome within a control zone, or enter the aerodrome traffic zone or traffic pattern:

   a) when the ceiling is less than 450 m (1500 ft); or

   b) when the ground visibility is less than 5 km.

3. VFR flights between sunset and sunrise, or such other period between sunset and sunrise as may be prescribed by the appropriate ATS authority, shall be operated in accordance with the conditions prescribed by such authority.

4. Unless authorized by the appropriate ATS authority, VFR flights shall not be operated:

   a) above FL 200;

   b) at transonic and supersonic speeds.

5. Authorization for VFR flights to operate above FL 290 shall not be granted in areas where a vertical separation minimum of 300 m (1000 ft) is applied above FL 290.

6. Except when necessary for take-off or landing, or except by permission from the appropriate authority, a VFR flight shall not be flown:

   a) over the congested areas of cities, towns or settlements or over an open-air assembly of persons at a height less than 300 m (1000 ft) above the highest obstacle within a radius of 600 m from the aircraft;

   b) elsewhere than as specified in 6. a), at a height less than 150 m (500 ft) above the ground or water.

7. Except where otherwise indicated in air traffic control clearances or specified by the appropriate ATS authority, VFR flights in level cruising flight when operated above 900 m (3 000 ft) from the ground or water, or a higher datum as specified by the appropriate ATS authority, shall be conducted at a cruising level appropriate to the track as specified in the tables of cruising levels in ICAO Annex 2: Rules of the Air Appendix 3.

8. VFR flights shall comply with the provisions of 3.6:

   a) when operated within Classes B, C and D airspace;

   b) when forming part of aerodrome traffic at controlled aerodromes; or

   c) when operated as special VFR flights

9. A VFR flight operating within or into areas, or along routes, designated by the appropriate ATS authority in accordance with ICAO Annex 2: Rules of the Air 3.3.1.2 c) or d) shall maintain continuous air-ground voice communication watch on the appropriate communication channel of, and report its position as necessary to, the air traffic services unit providing flight information service.

10. An aircraft operated in accordance with the visual flight rules which wishes to change to compliance with the instrument flight rules shall:

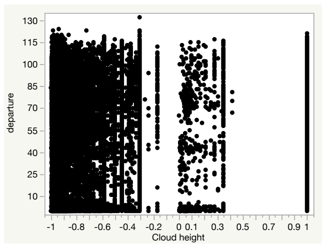
   a) if a flight plan was submitted, communicate the necessary changes to be effected to its current flight plan; or

   b) when so required by ICAO Annex 2: Rules of the Air 3.3.1.2, submit a flight plan to the appropriate air traffic services unit and obtain a clearance prior to proceeding IFR when in controlled airspace.

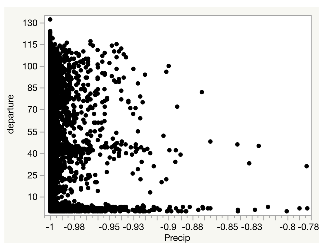
APPENDIX C
SENSITIVITY ANALYSIS

This section analyzes the sensitivity of the output to the input feature. Appendix C.1 explores changes in ATL's departure or arrival capacity according to the input variable value. On the other hand, Appendix C.2 examines the sensitivity of DEN-CLT delays minutes to input.

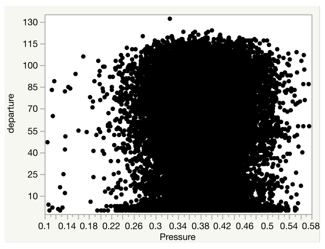
C.1 Sensitivity Analysis of Airport Capacity



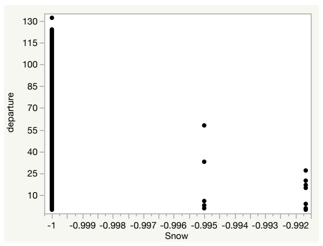
(a) Cloud Height



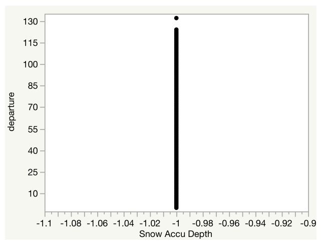
(b) Precipitation



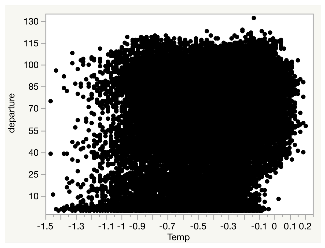
(c) Pressure



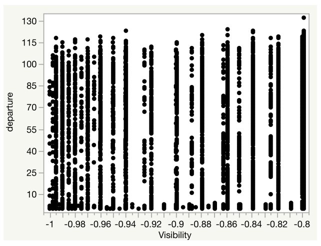
(d) Snow



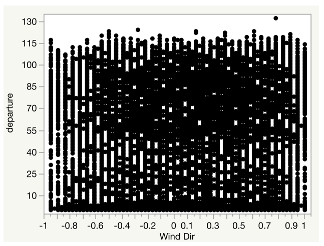
(e) Snow Depth



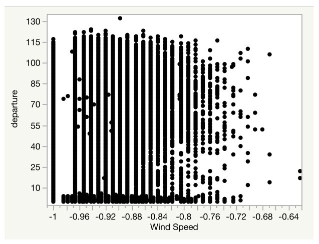
(f) Temperature



(g) Visibility

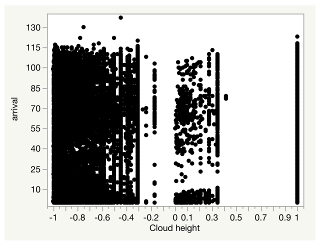


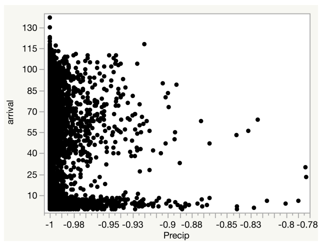
(h) Wind Direction

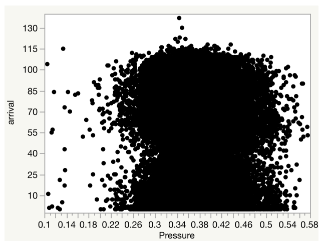


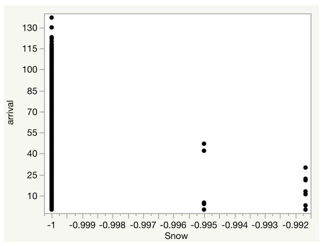
(i) Wind Speed

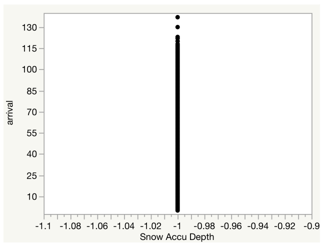
Figure C.1: Sensitivity Analysis of Departure Capacity

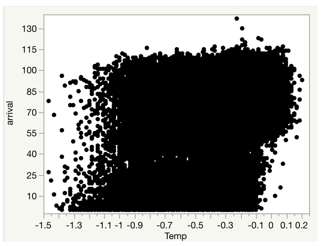

(a) Cloud Height

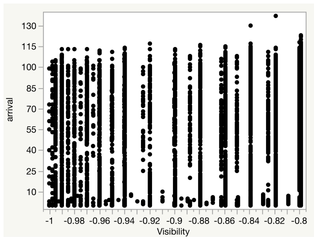

(b) Precipitation

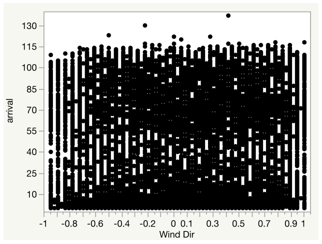

(c) Pressure


(d) Snow


(e) Snow Depth


(f) Temperature


(g) Visibility


(h) Wind Direction

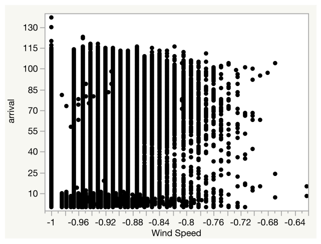
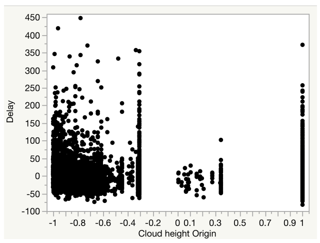
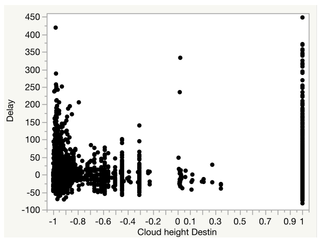

(i) Wind Speed

Figure C.2: Sensitivity Analysis of Arrival Capacity

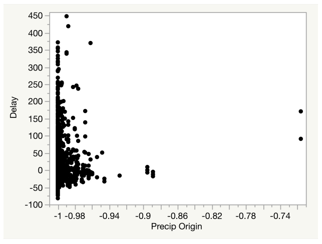
C.2 Sensitivity Analysis of Flight Delays



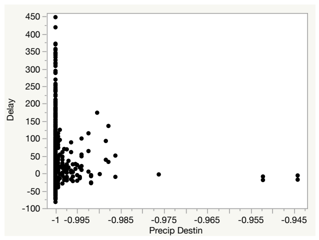
(a) Cloud Height of Origin



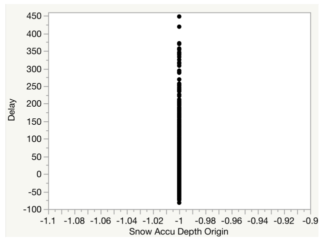
(b) Cloud Height of Destination



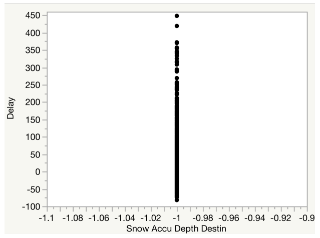
(c) Precipitation of Origin



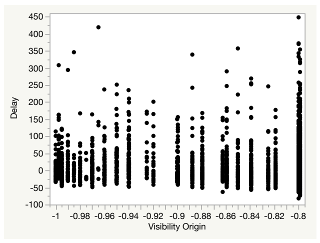
(d) Precipitation of Destination



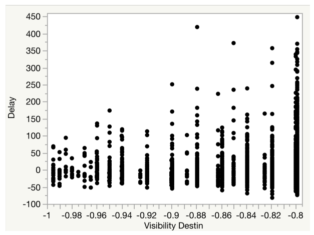
(e) Snow of Origin



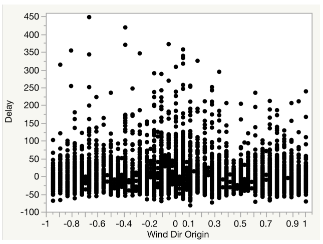
(f) Snow of Destination



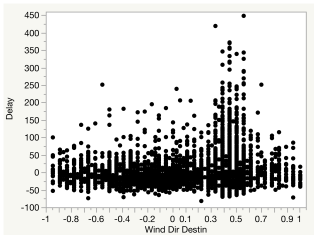
(g) Visibility of Origin



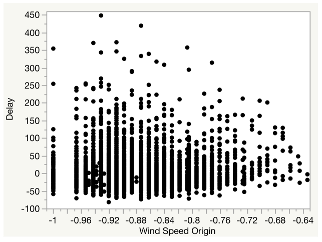
(h) Visibility of Destination



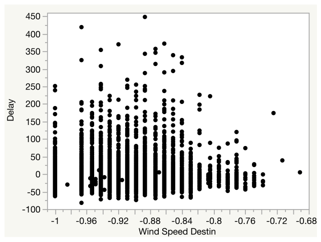
(i) Wind Direction of Origin



(j) Wind Direction of Destination



(k) Wind Speed of Origin



(l) Wind Speed of Destination

Figure C.3: Sensitivity Analysis of Delays

APPENDIX D
SPLITTING CRITERIA FOR CLASSIFICATION TREES

This section describes the criteria determining the goodness of a partition in the classification tree. A goodness function, $\theta(s,t)$, is defined as a function of the split, s, and the node, t. The split which can maximize $\theta(s,t)$ is selected as the optimal split [176]. The methods to define the Gini impurity criterion and Entropy, are introduced below. For the sake of simplicity, only the binary split and the binary classification are considered as shown in Figure D.1.

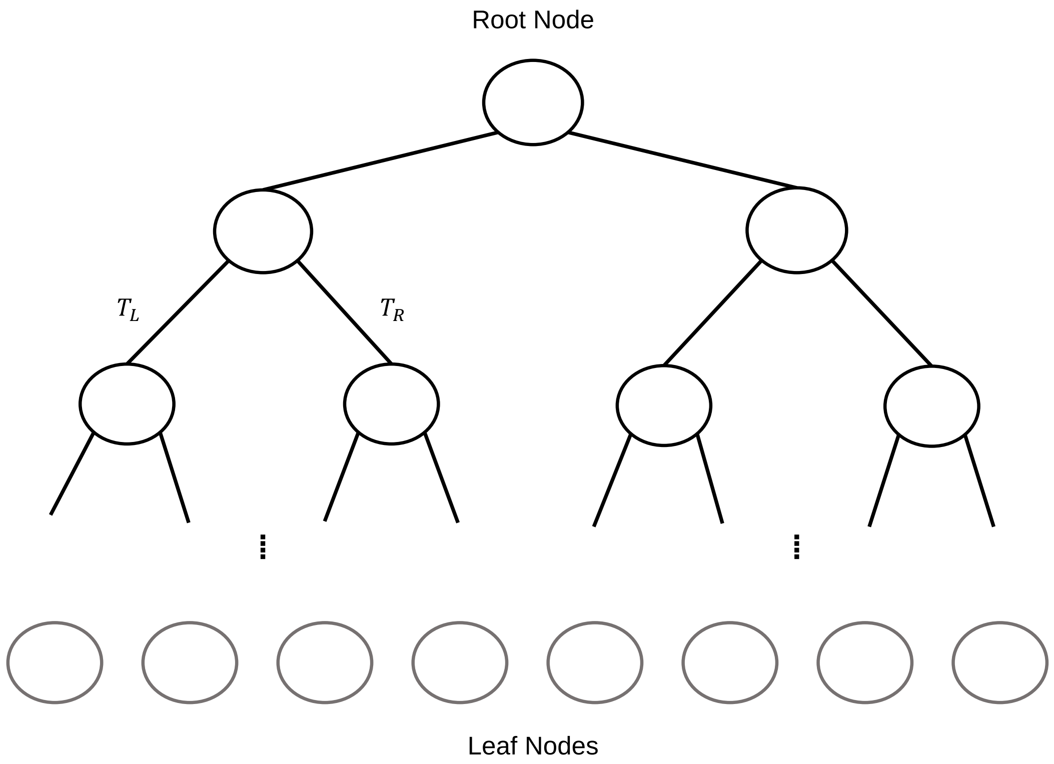


Figure D.1: Simple Schematic of Decision Tree

D.1 Gini Impurity Criterion [177]

Gini impurity criterion is a way to give more values to the candidate split that creates heterogeneous child nodes. Each split creates two child nodes, T_L and T_R, with N_L and N_R samples, respectively. π_L and π_R are defined as the ratio of samples in T_L and T_R among the total sample N.

$$\pi_L = \frac{N_L}{N}, \quad \pi_R = \frac{N_R}{N}$$

Let N_{jk} be the number of samples belonging to class $j \in \{0, 1\}$ in node $k \in \{L, R\}$. A vector $\mathbf{p} = (p_0, p_1)$ is used to represent the distribution of each class in the parent node. Similarly, p_{jk} represents the relative proportion of class $j \in \{0, 1\}$ in child node $k \in \{L, R\}$. The Gini index of each node is defined as

$$i(\mathbf{p}) = \sum_{j \neq l} p_j p_l = 1 - \sum_{j=0}^{1} p_j^2$$

The Gini index $i(p)$ is maximized when each class is evenly distributed on a node and minimized when on class dominate another. Then the Gini criterion for a parent node and two child nodes is

$$\theta(s,t) = \Delta i$$

$$= i(\mathbf{p}) - \pi_L i(\mathbf{p_L}) - \pi_R i(\mathbf{p_R})$$

$$= \pi_L \sum_{j=0}^{1} p_{jL}^2 + \pi_R \sum_{j=0}^{1} p_{jR}^2 - \sum_{j=0}^{1} p_j^2$$

where $\mathbf{p_L} = (p_{0L}, p_{1L})$ and $\mathbf{p_R} = (p_{0R}, p_{1R})$.

D.2 Entropy Criterion

Using the definition of the variables p_j above, the entropy criterion can be expressed as follows:

$$i(\mathbf{p}) = -\sum_{j=0}^{1} p_j \log p_j$$

The entropy is also maximized when the class distribution is uniform as is the case of the Gini criterion. Then the entropy criterion for a parent node and two child node, T_L and T_R is

$$\theta(s,t) = \Delta i$$

$$= i(\mathbf{p}) - \pi_L i(\mathbf{p_L}) - \pi_R i(\mathbf{p_R})$$

$$= \pi_L \sum_{j=0}^{1} p_{jL} \log p_{jL} + \pi_R \sum_{j=0}^{1} p_{jR} \log p_{jR} - \sum_{j=0}^{1} p_j \log p_j$$

Empirical studies have shown that the Gini and the entropy are about the same performance, but the entropy is slower because it requires logarithmic calculations.

APPENDIX E
FEATURE SELECTION

E.1 Individual Delays Prediction

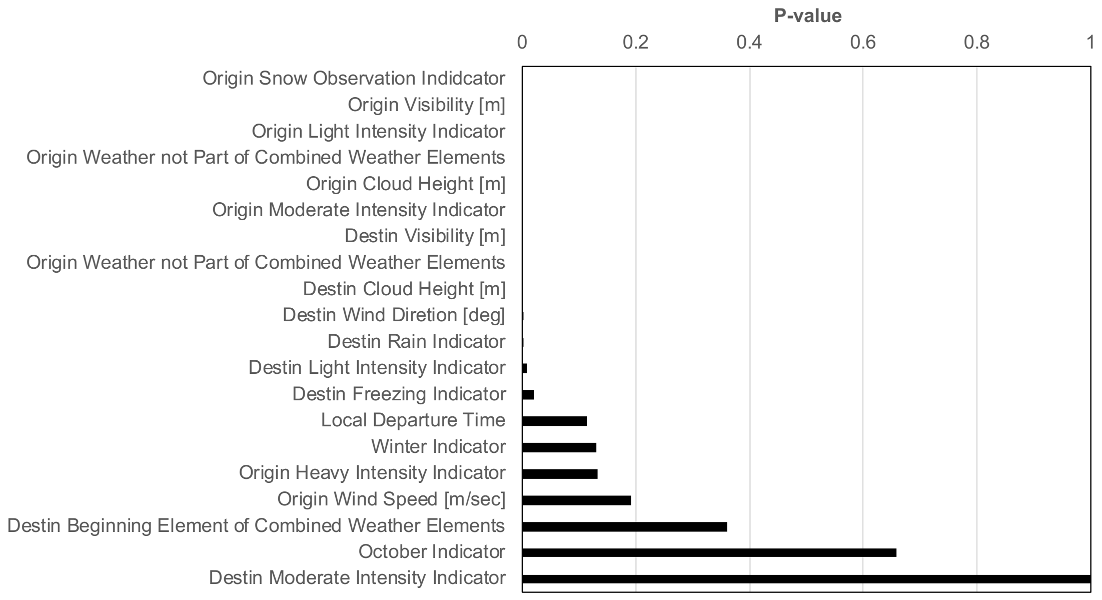


Figure E.1: p-value of Top 20 Variables

REFERENCES

[1] A. Sternberg, J. Soares, and CEFET/RJ. (Nov. 3, 2017). A Review on Flight Delay Prediction, arxiv:1703.06118v2 [cs.CY].

[2] Airline for America, *U.S. Airline Industry Review: Allocating Capital to Benefit Customers, Employees and Investors*, 2017.

[3] S. A. Morrison and C. Winston, "The effect of FAA expenditures on air travel delays," *Journal of Urban Economics*, vol. 63, pp. 669–678, 2008.

[4] FAA, "Aerospace Forecast Fiscal Years 2017-2037," Federal Aviation Administration, Tech. Rep., 2017.

[5] G. Flynn, A. Benkouar, and R. Christien, "Pessimistic Sector Capacity Estimation," Eurocontrol Experimental Centre, Tech. Rep., 2003.

[6] J. Martinez, A. Trani, and P. Ioannou, "Modeling Airside Airport Operations Using General-Purpose, Activity-Based, Discrete-Event Simulation Tools," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1744, pp. 65–71, 2001.

[7] FAA, "FACT3: Airport Capacity Needs in the National Airspace System," Federal Aviation Administration, Tech. Rep., 2015.

[8] International Civil Aviation Organization, *Manual on Collaborative Air Traffic Flow Management*, 2nd Edition, 2014, ISBN: 978-92-9249-418-6.

[9] US-India Aviation Cooperation Program, *Introduction to Air Traffic Flow Management (atfm)*.

[10] L. Fisher, "Evaluating Airfield Capacity," TRANSPORTATION RESEARCH BOARD, Tech. Rep., 2012.

[11] M. O. Ball and G. Lulli, "Ground Delay Programs: Optimizing over the Included Flight Set Based on Distance," *Air Traffic Control Quarterly*, vol. 12, no. 1, pp. 1 –25, 2004.

[12] T. Vossen, M. Ball, R. Hoffman, and M. Wambsganss, "A General Approach to Equity in Traffic Flow Management and Its Application to Mitigating Exemption Bias in Ground Delay Programs," *Air Traffic Control Quarterly*, vol. 11, no. 4, pp. 277–292, 2003.

[13] FAA Collaborative Decision Making. (). What′ s different about a CDM GDP? Available at https://cdm.fly.faa.gov/wp-content/list_yo_files_user_folders/cdm_editor/cdm_arch_train_gdpe/GDPS.pdf.

[14] J. I. Robinson and M. Kamgarpour, “Benefits of Continuous Descent Operations in High-Density Terminal Airspace Considering Scheduling Constraints,” in *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, Fort Worth, Texas, USA, 2010.

[15] J.-P. B. Clarke, N. T. Ho, L. Ren, J. A. Brown, K. R. Elmer, K.-O. Tong, and J. K. Wat, “Continuous Descent Approach: Design and Flight Test for Louisville International Airport,” *JOURNAL OF AIRCRAFT*, vol. 41, no. 5, pp. 1054–1066, 2004.

[16] I. Wilson and F. Hafner, “Benefit assessment of using continuous descent approaches at Atlanta,” in *The 24th IEEE/AIAA Digital Avionics Systems Conference*, Washington, DC, USA, 2005.

[17] L. A. Weitz, J. E. Hurtado, and F. J. L. Bussink, “Increasing Runway Capacity for Continuous Descent Approaches Through Airborne Precision Spacing,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, California, USA, 2005.

[18] J. A. Filar, P. Manyem, and K. White, “How Airlines and Airports Recover from Schedule Perturbations: A Survey,” *Annals of Operations Research*, vol. 108, pp. 315–333, 2001.

[19] M. Janic, “A MODEL OF AIR TRAFFIC CONTROL SECTOR CAPACITY BASED ON AIR TRAFFIC CONTROLLER WORKLOAD,” *Transportation Planning and Technology*, vol. 20, pp. 311–335, 1997.

[20] P. Brooker, “Control Workload, Airspace Capacity and Future Systems,” *Human Factors and Aerospace Safety*, vol. 3, no. 1, pp. 1–23, 2003.

[21] A. Majumdar, W. Y. Ochieng, J. Bentham, and M. Richards, “En-route sector capacityestimation methodologies: An international survey,” *Journal of Air Transport Management*, vol. 11, pp. 375–387, 2005.

[22] T. Guest, “A Matter of Time: Air Traffic Delay in Europe,” Eurocontrol, Tech. Rep., 2007.

[23] M. Jetzki, “The propagation of air transport delays in Europe,” PhD thesis, Rwth Aachen University, 2009.

[24] BTS. (2016). TranStats, U.S. Department of Transportation.

[25] M. Ball, C. Barnhart, M. Dresner, M. Hansen, K. Neels, A. Odoni, E. Peterson, L. Sherry, A. Trani, and B. Zou, "Total Delay Impact Study: A Comprehensive Assessment of the Costs and Impacts of Flight Delay in the United States," The National Center of Excellence for Aviation Operations Research, Tech. Rep., 2010.

[26] T. K. Simic and O. Babic, "Airport traffic complexity and environment efficiency metrics for evaluation of atm measures," *Journal of Air Transport Management*, vol. 42, pp. 260–271, 2015.

[27] T. Pejovic, R. B. Noland, V. Williams, and R. Toumi, "A tentative analysis of the impacts of an airport closure," *Journal of Air Transport Management*, vol. 15, pp. 241–248, 2009.

[28] Y. Xu, R. Dalmau, and X. Prats, "Maximizing airborne delay at no extra fuel cost by means of linear holding," *Transportation Research Part C*, vol. 81, pp. 137–152, 2017.

[29] International Civil Aviation Organization and Commercial Aviation Safety Team, *Phase of flight, Definitions and usage notes*, 1.3, 2013.

[30] T. Arts, C. Asma, P. Corieri, N. D. Pascale, C. Dobre, T. Kirmse, and M. Riethmuller, *How does an airplane fly? Basic principles Environmental and safety issues*. the von Karman Institute for Fluid Dynamics, 2010, ISBN: 978-2-87516-013-3.

[31] U.S. Department of Transportation, *Tarmac delays*, Mar. 13, 2018.

[32] US Federal Aviation Administration, *Air traffic management glossary of terms*, 2018.

[33] K. B. Laskey, N. Xu, and C.-H. Chen. (2012). Propagation of delays in the national airspace system.

[34] M. Terrab and A. R. Odoni, "Strategic Flow Management for Air Traffic Control," in *IEEE International Conference on Systems, Man and Cybernetics*, 1992.

[35] A. Jacquillat, "A Queuing Model of Airport Congestion and Policy Implications at JFK and EWR," Master's thesis, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, 2012.

[36] L. Zonglei, W. Jiandong, and Z. Guansheng, "A New Method to Alarm Scale of Flight Delay Based on Machine Learning," in *Proceedings of International Symposium on Knowledge Acquisition and Modeling*, 2008.

[37] J. Gentry, K. Duffy, and W. J. Swedish, "Airport Capacity Profiles," Federal Aviation Administration, Tech. Rep., 2014.

[38] E. P. Gilbo, "Optimizing Airport Capacity Utilization in Air Traffic Flow Management Subject to Constraints at Arrival and Departure Fixes," *IEEE Transactions on Control Systems Technology*, vol. 5, no. 5, pp. 490–503, 1997.

[39] Eurocontrol, "European aviation in 2040: Challenges of growth," EUROCONTROL, Tech. Rep., 2018.

[40] T. Cheng, D. Cui, and P. Cheng, "Data mining for air traffic flow forecasting: A hybrid model of neural network and statistical analysis," in *Intelligent Transportation Systems*, 2003.

[41] Y.-S. Jeong, Y.-J. Byon, M. M. Castro-Neto, and S. M. Easa, "Supervised Weighting-Online Learning Algorithm for Short-Term Traffic Flow Prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, pp. 1700–1707, 4 2013.

[42] S.-L. Tien, S. Roy, C. Taylor, C. Wanke, and R. Dhal, "Evaluation of an Airport Capacity Prediction Model for Strategizing Air Traffic Management," in *American Meteorological Society 95th Annual Meeting*, 2015.

[43] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic Flow Prediction With Big Data: A Deep Learning Approach," *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS*, vol. 16, no. 2, 2015.

[44] J. Kim, "An Agent-Based Model for Airline Evolution, Competition and Airport Congestion," PhD thesis, Virginia Polytechnic Institute and State University, 2005.

[45] D. Yuan, "Flight Delay-Cost Simulation Analysis and Airline Schedule Optimization," PhD thesis, RMIT University, Victoria, Australia, 2007.

[46] DOT. (2016). February 2016 On-Time Performance Up From Previous Year, January 2016.

[47] CNN. (Sep. 11, 2001). All flights stopped nationwide. Available at `http://www.cnn.com/2001/TRAVEL/NEWS/09/11/faa.airports`, (visited on 07/19/2018).

[48] Federal Aviation Administration. (). FAQ: Weather Delay.

[49] Z. Hua, X. Zhang, and X. Xu, "ASYMMETRIC SUPPORT VECTOR MACHINE FOR THE CLASSIFICATION PROBLEM WITH ASYMMETRIC COST OF MISCLASSIFICATION," *International Journal of Innovative Computing, Information and Control*, vol. 6, no. 12, pp. 5597–5608, 2010.

[50] B. Zadrozny, J. Langford, and N. Abe, “Cost-Sensitive Learning by Cost-Proportionate Example Weighting,” in *Proceedings of the Third IEEE International Conference on Data Mining*, IEEE Computer Society, 2003.

[51] C. Elkan, “The foundations of cost-sensitive learning,” in *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI’01)*, Aug. 4, 2001.

[52] D. D. Margineantu, “On class-probability estimates and cost-sensitive evaluation of classiers,” in *Workshop on Cost-Sensitive Learning, International Conference on Machine Learning*, 2000.

[53] S. Choi, Y. J. Kim, S. Briceno, and D. Mavris, “Prediction of weather-induced airline delays based on machine learning,” in *Digital Avionics Systems Conference, 2016 IEEE/AIAA 35th*, 2016.

[54] ——, “Cost-sensitive prediction of airline delays using machine learning,” in *Digital Avionics Systems Conference (DASC), 2017 IEEE/AIAA 36th*, 2017.

[55] R. Kicinger, C. Cross, T. Myers, J. Krozel, C. Mauro, and D. Kierstead, “Probabilistic Airport Capacity Prediction Incorporating the Impact of Terminal Weather,” in *AIAA Guidance, Navigation, and Control Conference*, 2011.

[56] W. J. Swedish, “Upgraded FAA Airfield Capacity Model. Volume 1. Supplemental User’s Guide,” The MITRE Corporation, Tech. Rep., 1981.

[57] C. T. Ball, *Model user’s manual for airfield capacity and delay models*, 1976.

[58] R. Kicinger, J. Krozel, M. Steiner, and J. Pinto, “Airport Capacity Prediction Integrating Ensemble Weather Forecasts,” in *Infotech@Aerospace Conferences*, 2012.

[59] R. Kicinger, J.-T. Chen, M. Steiner, and J. Pinto, “Airport Capacity Prediction with Explicit Consideration of Weather Forecast Uncertainty,” *Journal of Air Transportation*, vol. 24, no. 1, 2016.

[60] P. Subramanian, “A simulation study to investigate runway capacity using taam,” Master’s thesis, Embry-Riddle Aeronautical University, 2002.

[61] FAA, *Simmod Manual: How Simmod Works*, FAA Capacity Modeling & Analysis Group.

[62] A. R. Odoni, J. Bowman, D. Delahaye, J. J. Deyst, E. Feron, R. J. Hansman, K. Khan, J. K. Kuchar, N. Pujet, and R. W. Simpson, “Existing and required modeling capabilities for evaluating atm systems and concepts,” INTERNATIONAL CEN-

TER FOR AIR TRANSPORTATION and MASSACHUSETTS INSTITUTE OF TECHNOLOGY, Modeling Research Under NASA/AATT, 1997.

[63] P. C. Kuzminski, "An improved runwaysimulator - simulation for runway system capacity estimation," in *Integrated Communications, Navigation and Surveillance Conference (ICNS)*, 2013.

[64] LeighFisher, L. Brown, N. R.C.U.T. R. Board, A. C. R. Program, and U. S.F. A. Administration, *Evaluating Airfield Capacity*, ser. ACRP report. Transportation Research Board, 2012, ISBN: 9780309258739.

[65] A. Kim and M. Hansen, "Validation of Runway Capacity Models," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2177, no. pp.69-77, 2010.

[66] TransSolutions, F. Consulting, I. Harris Miller Miller & Hanson, and J. Rakas, "Defining and Measuring Aircraft Delay and Airport Capacity Thresholds," Airport Cooperative Research Program, Tech. Rep., 2014.

[67] J. N. Barrer, P. Kuzminski, and W. J. Swedish, "Analyzing the runway capacity of complex airports," The MITRE Corporation, Tech. Rep., 2005, Available at `https://www.mitre.org/publications/technical-papers/analyzing-the-runway-capacity-of-complex-airports`.

[68] J. C. Jones, R. DeLaura, M. Pawlak, S. Troxel, and N. Underhill, "Predicting & quantifying risk in airport capacity profile selection for air traffic management," in *Twelfth USA/Europe Air Traffic Management Research and Development Seminar (ATM2017)*, 2017.

[69] FAA. (2016). runwaySimulator Airport Capacity Model. Available at `https://www.faa.gov/airports/planning_capacity/runwaysimulator/`.

[70] ——, (2017). The Aviation System Performance Metrics (ASPM).

[71] *Pilot and Air Traffic Control Guide to Wake Turbulence*, FAA, 2015.

[72] ——, "NY/NJ/PHL Airspace Redesign Final Environmental Impact Statement (FEIS) - Appendix A. National Airspace System Overview," US DOT, FINAL ENVIRONMENTAL IMPACT STATEMENT, 2007.

[73] S. L. M. Mockaday and A. K. Kanafani, "Developments in Airport Capacity Analysis," *Transportation Research*, vol. 8, pp. 171 –180, 1974.

[74] S. D. Thompson, "Terminal Area Separation SStandard: Historical Development, Current Standards, and Processes for Change," Lincoln Laboratory, MIT, Lexington, Massachusetts, Project Report, 1997.

[75] *Advisory Circular 150/5300-13A - Airport Design*, U.S. Department of Transportation, 2012.

[76] Y. J. Kim, S. Choi, S. Briceno, and D. Mavris, "A Deep Learning Approach to Flight Delay Prediction," in *IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, 2016.

[77] S. Kim and D. H. Shin, "Forecasting short-term air passenger demand using big data from search engine queries," *Automation in Construction*, vol. 70, pp. 98–108, 2016.

[78] J. Ke, H. Zheng, H. Yang, and X. M. Chen. (Jun. 20, 2017). Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach, arxiv:1706.06279v1 [cs.LG].

[79] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, 4 1943.

[80] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[81] M. Minsky and S. A. Papert, *Perceptrons: An Introduction to Computational Geometry*. The MIT Press, 1969, ISBN: 0262631113.

[82] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 1, pp. 318 –362, 1986.

[83] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, IEEE, 1998.

[84] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, pp. 1527–1554, 2006.

[85] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *In Proceedings Advances in Neural Information Processing Systems*, vol. 19, 2006, pp. 153–160.

[86] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale

Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211 –252, 2015.

[87] G. v. Zitzewitz, "Survey of neural networks in autonomous driving," in *ADVANCED SEMINAR SS 2017: SURVEY OF NEURAL NETWORKS IN AUTONOMOUS DRIVING*, 2017.

[88] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Neural Information Processing Systems*, 2012, pp. 1106–1114.

[89] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. (Sep. 17, 2014). Going deeper with convolutions, arxiv:1409.4842v1 [cs.CV].

[90] Q. Le and B. Zoph. (May 17, 2017). Using machine learning to explore neural network architecture. Available at `https://ai.googleblog.com/2017/05/using-machine-learning-to-explore.html`.

[91] K. He, X. Zhang, S. Ren, and J. Sun. (Dec. 10, 2015). Deep residual learning for image recognition, arxiv:1512.03385v1 [cs.CV].

[92] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 39, no. 4, pp. 664–676, 2017.

[93] H. Yan, Y. J. Jiang, J. Zhen, C. Peng, and Q. Li, "A multilayer perceptron-based medical decision support system for heart disease diagnosis," *Expert Systems with Applications*, vol. 30, pp. 272–281, 2006.

[94] M. W. Gardner and S. R. Dorling, "Artificial Neural Network (The Multilayer Perceptron) - A Review of Applications in the Atmospheric Sciences," *Atmospheric Environment*, vol. 32, no. 14/15, pp. 2627–2636, 1998.

[95] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT' 2010*, Springer, Sep. 30, 2010, pp. 177 –186.

[96] W. D. Mulder, S. Bethard, and M.-F. Moens, "A survey on the application of recurrent neural networks to statistical language modeling," *Computer Speech and Language*, vol. 30, pp. 61 –98, 2015.

[97] T. Mikolov, M. Karafiat, L. Burget, J. H. Cernocky, and S. Khudanpur, "Recurrent Neural Network Based Language Model," in *Interspeech*, 2010.

[98] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith. (2015). Transition-Based Dependency Parsing with Stack Long Short-Term Memory, arxiv:1505.08075v1 [cs.CL].

[99] Y. Bengio, P. Simard, and P. Frasconi, "Learning Long-Term Dependencies with Gradient Descent is Difficult," *IEEE TRANSACTIONS ON NEURAL NETWORKS*, vol. 5, no. 2, pp.157 –166, 1994.

[100] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. (Sep. 3, 2014). Learning phrase representations using rnn encoderdecoder for statistical machine translation, arxiv:1406.1078 [cs.CL].

[101] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. (Dec. 11, 2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, arxiv:1412.3555v1 [cs.NE].

[102] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, pp. 1735 –1780, 1997.

[103] A. Graves, A.-r. Mohamed, and G. Hinton. (Mar. 22, 2013). SPEECH RECOGNITION WITH DEEP RECURRENT NEURAL NETWORKS arxiv:1303.5778v1 [cs.NE].

[104] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400 –407, 1951.

[105] S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le. (Feb. 24, 2018). Don't Decay the Learning Rate, Increase the Batch Size, arxiv:1711.00489v2 [cs.LG].

[106] I. Loshchilov and F. Hutter. (May 3, 2017). SGDR: Stochastic Gradient Descent with Warm Restarts, arxiv:1608.03983v5 [cs.LG].

[107] S. Zagoruyko and N. Komodakis. (Jun. 14, 2017). Wide Residual Networks, arxiv:1605.07146v4 [cs.CV].

[108] S. Ruder. (Jun. 15, 2017). An overview of gradient descent optimization algorithms, arxiv:1609.04747v2 [cs.LG].

[109] N. Qian, "On the Momentum Term in Gradient Descent Learning Algorithms," *Neural networks: the official journal of the International Neural Network Society*, vol. 12, no. 1, pp. 145 –151, 1999.

[110] D. P. Kingma and J. L. Ba. (Jan. 30, 2017). ADAM: A Method for Stochastic Optimization, arxiv:1412.6980v9 [cs.LG].

[111] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, Y. Bengio, Ed., pp. 1929 –1958, 2014.

[112] Z. C. Lipton, J. Berkowitz, and C. Elkan. (Oct. 17, 2015). A Critical Review of Recurrent Neural Networks for Sequence Learning, arxiv:1506.00019v4 [cs.LG].

[113] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed. draft. 2017.

[114] R. L. Stratonovich, "Conditional markov processes," *Theory of Probability & Its Applications*, vol. 5, no. 2, pp. 156–178, 1960.

[115] A. Graves, *Supervised sequence labelling with Recurrent Neural Networks*. Springer, 2012, vol. 385.

[116] N. Friedman and J. H. Halpern. (2013). A qualitative markov assumption and its implications for belief change.

[117] FAA. (2018). Aviation system performance metrics (ASPM). Available at `https://aspm.faa.gov/apm/sys/main.asp`.

[118] NOAA. (2018). Integrated surface database (ISD). Available at `https://www.ncdc.noaa.gov/isd`.

[119] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang. (Dec. 3, 2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems, arxiv:1512.01274v1 [cs.DC].

[120] Eurocontrol and Federal Aviation Administration, "U.S./europe comparison of atm-related operational performance," Eurocontrol and Federal Aviation Administration, Tech. Rep., 2009.

[121] R. Girshick, J. Donahue, T. Darrell, and J. Malik. (Oct. 22, 2014). Rich feature hierarchies for accurate object detection and semantic segmentation, arxiv:1311.2524v5 [cs.CV], Tech report (v5).

[122] L. Y. Pratt, "Discriminability-based transfer between neural networks," in *NIPS Conference: Advances in Neural Information Processing Systems 5*, 1993, pp. 204–211.

[123] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" In *Advances in Neural Information Processing Systems 27*, 2014, pp. 3320 –3328.

[124] A. McAfee and E. Brynjolfsson, “Big Data: The Management Revolution,” *Harvard Business Review*, vol. 90, no. 10, pp. 60–68, 2012.

[125] A. Klein, R. Jehlen, and D. Liang, “Weather Index with Queuing Component for National Airspace System Performance Assessment,” in *7th USA/Europe ATM R&D Seminar*, 2007.

[126] A. Klein, C. Craun, and R. S. Lee, “Airport Delay Prediction Using Weather-Impacted Traffic Index (WITI) Model,” in *Preceedings of the Digital Avionics Systems Conference(DASC) 29th*, 2010.

[127] A. Klein, T. MacPhail, S. Kavoussi, D. Hickman, M. Phaneuf, R. S. Lee, and D. Simenauer, “NAS WEATHER INDEX: QUANTIFYING IMPACT OF ACTUAL AND FORECAST EN-ROUTE AND SURFACE WEATHER ON AIR TRAFFIC,” in *14th Conference on Aviation, Range and Aerospace Meteorology*, 2009.

[128] M. Abdel-Aty, C. Lee, Y. Bai, X. Li, and M. Michalak, “Detecting periodic patterns of arrival delay,” *Journal of Air Transport Management*, 2007.

[129] S. Pathomsiri, A. Haghani, M. Dresner, and R. J. Windle, “Impact of undesirable outputs on the productivity of US airports,” *Transportation Research Part E*, vol. 44, pp. 235–259, 2008.

[130] Y. Tu, M. O. Ball, and W. S. Jank, “Estimating Flight Departure Delay Distributions-A Statistical Approach with Long-Term Trend and Short-Term Pattern,” *Journal of the American Statistical Association*, vol. 103, no. 481, pp. 112–125, 2008.

[131] Y. Bai, “Analysis of Aircraft Arrival Delay And Airport On-time Performance,” Master’s thesis, University of Central Florida, 2006.

[132] Y. J. Kim, O. J. Pinon, and D. N. Mavris, “Parallel Simulation of Agent-Based Model for Air Traffic Network,” in *AIAA Modeling and Simulation Technologies Conference*, 2015.

[133] G. Marcus. (Jan. 2, 2018). Deep learning: A critical appraisal, arxiv:1801.00631 [cs.AI].

[134] S. Sabour, N. Frosst, and G. E. Hinton. (Oct. 26, 2017). Dynamic routing between capsules.

[135] J. J. Rebollo and H. Balakrishnan, “Characterization and prediction of air traffic delays,” *Transportation Research Part C: Emerging Technologies*, vol. 44, pp. 231–241, 2014.

[136] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[137] N. Williams, S. Zander, and G. Armitage, "A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 5, 2006.

[138] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[139] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[140] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer New York, 2013.

[141] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2007.

[142] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[143] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[144] V. Garca, J. Snchez, R. Mollineda, J. Sotoca, and A. R., "The class imbalance problem in pattern classification and learning," in *II Congreso Espanol de Informtica, CEDI 2007*, 2007, pp. 978–984.

[145] N. V. Chawla, "C4.5 and imbalanced data sets: Investigating the effect of sampling method, probabilistic estimate, and decision tree structure," in *Proceedings of the ICML, Workshop on Learning from Imbalanced Datasets II*, Washington DC, 2003.

[146] R. C. Bhagat and S. S. Patil, "Enhanced smote algorithm for classification of imbalanced big-data using random forest," in *Proceedings of the Advance Computing Conference (IACC)*, 2015.

[147] BTS. (2016). Airline on-time performance data, US DOT.

[148] A. Smith, N. Lott, and R. Vose, "The integrated surface database: Recent developments and partnerships," *Bulletin of the American Meteorological Society*, vol. 92, pp. 704–708, 2011.

[149] S. Allan, S. Gaddy, and J. Evans, "Delay Causality and Reduction at the New York City Airports Using Terminal Weather Information Systems," Lincoln Laboratory, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, Tech. Rep., 2001.

[150] (2016). Flightstats. Available at `https://developer.flightstats.com/`.

[151] (2016). World weather online. Available at `http://developer.worldweatheronline.com/api/`.

[152] J. H. Friedman, R. Tibshirani, and T. Hastie, *The Elements of Statistical Learning*. Springer, 2008.

[153] D. M. W. Powers, "Evaluation: From precision, recall and f-measure to roc, informedness, markedness & correlation," *Journal of Machine Learning Technologies*, vol. 2, pp. 37 –63, 1 2011.

[154] scikit-learn. (2017). Receiver Operating Characteristic (ROC). Available at `http://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html`.

[155] M. Pechenizkiy, "The impact of feature extraction on the performance of a classifier: Knn, nave bayes and c4.5," in *18th CSCSI Conference on Artificial Intelligence AI05*, Springer Verlag, 2005, pp. 268–279.

[156] US Federal Aviation Administration, "AC 25-31 - Takeoff Performance Data for Operations on Contaminated Runways," Advisory Circular, Dec. 22, 2015.

[157] S. C. Grandeau, "The processes of airline operational control," Master's thesis, Massachusetts Institute of Technology, 1995.

[158] M. Salem, S. Taheri, and J.-S. Yuan, "An experimental evaluation of fault diagnosis from imbalanced and incomplete data for smart semiconductor manufacturing," *Big Data and Cognitive Computing*, vol. 2, p. 30, 4 Sep. 21, 2018.

[159] J. Jakeman, M. Eldred, and D. Xiu, "Numerical approach for quantification of epistemic uncertainty," *Journal of Computational Physics*, vol. 229, pp. 4648 –4663, 2010.

[160] B. J. Johnson, "An uncertainty quantification and management methodology to support rework decisions in multifidelity aeroelastic load cycles," PhD thesis, Georgia Institute of Technology, 2017.

[161] B. A. Lockwood, M. Anitescu, and D. J. Mavriplis, "Mixed aleatory/epistemic uncertainty quantification for hypersonic flows via gradient-based optimization and

surrogate models," in *50th AIAA Aerospace Sciences Meeting and Exhibit, Nashville, TN*, 2012.

[162] R. Nau. (2014). Review of basic statistics and the simplest forecasting model: The sample mean. Available at people.duke.edu/ rnau/forecasting.htm.

[163] N. V. Chawla, "Data mining for imbalanced datasets: An overview," in *Data Mining and Knowledge Discovery Handbook*. 2010, ch. pp. 875-886.

[164] K. Gopalakrishnan and H. Balakrishnan, "A comparative analysis of models for predicting delays in air traffic networks," in *Twelfth USA/Europe Air Traffic Management Research and Development Seminar (ATM2017)*, 2017.

[165] D. Joo, T. Hong, and I. Han, "The neural network models for IDS based on the asymmetric costs of false negative errors and false positive errors," *Expert Systems with Applications*, vol. 25, pp. 69–75, 2003.

[166] F. Yang, H. Wang, H. Mi, C. Lin, and W. Cai, "Using random forest for reliable classification and cost-sensitive learning for medical diagnosis," *BMC Bioinformatics*, vol. 10, S22 2009.

[167] S. Uguroglu, J. Carbonell, M. Doyle, and R. Biederman, "Cost-Sensitive Risk Stratification in the Diagnosis of Heart Disease," in *The Twenty-Fourth Innovative Applications of Artificial Intelligence Conference*, 2012.

[168] C. Sammut and G. I. Webb, Eds., *Encyclopedia of Machine Learning*. Springer, 2011, ISBN: 978-0-387-30768-8.

[169] P. Domingos, "MetaCost: A general method for making classifiers cost sensitive," in *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining*, 1999, pp.155–164.

[170] C. X. Ling and V. S. Sheng, "Cost-Sensitive Learning and the Class Imbalance Problem," *Encyclopedia of Machine Learning*, 2008.

[171] University of California, Irvine. (Feb. 16, 1999). KDD Cup 1998 Data. Available at `https://kdd.ics.uci.edu/databases/kddcup98/kddcup98.html`.

[172] M. Collier and S. Smith, "The flight dispatch process," American Airlines, Cross Polar Working Group, 2017, Available at `https://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/systemops/ato_intl/documents/cross_polar/CPWG23/CPWG23_Brf_AAL_Dispatch_Process.pdf`.

[173] G. Kim, “Who is the dispatcher?” NASA Ames Research Center, Emergent Aircraft Systems and the Dispatcher, Mar. 18, 2018.

[174] FAA, “NOTAMs, Getting back to basics,” 2018, Available at `https://www.faa.gov/about/initiatives/notam/media/NOTAM_101_Primer.pdf`.

[175] *Annex 2 Rules of the Air*, International Standards, ICAO, 2005.

[176] L. Breiman, “Technical note: Some properties of splitting criteria,” *Machine Learning*, vol. 24, pp. 41 –47, 1996.

[177] Y.-S. Shih, “Families of splitting criteria for classification trees,” *Statistics and Computing*, vol. 9, pp. 309 –105, 1999.

[178] D. F. Williamson, R. A. Parker, and J. S. Kendrick, “The box plot: A simple visual method to interpret data,” *Annals of Internal Medicine*, vol. 110, no. 11, pp. 916 –921, Jun. 1, 1989.