CEFET/RJ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO

ANALYZING FLIGHT DELAY PREDICTION UNDER CONCEPT DRIFT

Lucas Giusti Tavares

Dissertation submitted to the Postgraduate Program in of the Federal Center for Technological Education of Rio de Janeiro, CEFET/RJ, as partial fulfillment of the requirements for the degree of master.

Advisors:
Jorge de Abreu Soares e Eduardo Soares Ogasawara (Co-advisor)

Rio de Janeiro,

October 2021

Analyzing Flight Delay Prediction Under Concept Drift

Dissertation submitted to the Postgraduate Program in of the Federal Center for Technological Education of Rio de Janeiro, CEFET/RJ, as partial fulfillment of the requirements for the degree of master.

Lucas Giusti Tavares

Rio de Janeiro,

October 2021

Ficha catalográfica elaborada pela Biblioteca Central do CEFET/RJ

T231 Tavares, Lucas Giusti
Analyzing flight delay prediction under concept drift / Lucas Giusti Tavares — 2021.
32f : il. (algumas color.), enc.

Dissertação (Mestrado) Centro Federal de Educação Tecnológica Celso Suckow da Fonseca, 2021.
Bibliografia : f. 27-32
Orientador: Jorge de Abreu Soares
Coorientador: Eduardo Soares Ogasawara

1. Transporte aéreo. 2. Transporte – Tabela de horário. 3. Aeronáutica – Voos. 4. Aeronáutica comercial. 5. Aprendizado de máquina. I. Soares, Jorge de Abreu (Orient.). II. Ogasawara, Eduardo Soares (Coorient.). III. Título.

CDD 387.7

Elaborada pela bibliotecária Tania Mello – CRB/7 nº 5507/04

DEDICATION

Dedico esta dissertação a todos que contribuíram para a produção e disseminação de conhecimento científico.

"Na ciência, a autoridade de milhares de opiniões não vale tanto quanto uma pequena fagulha de razão de um homem." - Galileo Galilei

ACKNOWLEDGMENTS

Aos orientadores desta dissertação, serei eternamente grato pelo suporte, ensinamentos e exemplos oferecidos durante todo o curso. Sem dúvida fizeram a diferença na minha carreia e na vida.

Aos meus pais, agradeço pela dedicação em me prover educação de qualidade e dar suporte para me desenvolver como profissional e pessoa, o que possibilita conquistas como essa.

À minha companheira Nadinne Holanda, meu primo Renato Del Rei e colega de UERJ, Valter Moreno, por apresentarem e estimularem minha evolução nas áreas de computação e ciências exatas.

Agrageço também pelo apoio da Coordenacão de Aperfeicionamento de Pessoal de Nível Superior - Brasil(CAPES), do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e da Fundacão de Amparo 'a Pesquisa do Estado do Rio de Janeiro (FAPERJ) no desenvolvimento dessa dissertação.

RESUMO

Analyzing Flight Delay Prediction Under Concept Drift

Lucas Giusti Tavares

Advisors:
Jorge de Abreu Soares e Eduardo Soares Ogasawara (Co-advisor)

Resumo da Dissertação submetida ao Programa de Pós-graduação em Ciência da Computação do Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ como parte dos requisitos necessários à obtenção do grau de mestre.

Atraso é um dos indicadores mais críticos para sistemas de transporte aéreo. Atrasos de voos impõem um desafio que impacta qualquer sistema de transporte aéreo. Nesse contexto, a previsão de voos atrasados pode ser uma ferramenta essencial para tratar desse problema de forma efetiva. A presente dissertação investiga o desempenho de previsão das estratégias ativa e passiva de tratamento de drift na aviação em diferentes escalas. Duas diferentes escalas foram consideradas: *system-based* (SB) e *airport-based* (AB). Na SB, todos os aeroportos no sistema aéreo são considerados. De maneira oposta, no AB, cada aeroporto é estudado separadamente. Especificamente, esse trabalho propôs e respondeu duas perguntas de pesquisa: (i) Como estratégias de tratamento de drift influenciam o desempenho da previsão de atrasos?; e (ii) Diferentes escalas podem influenciar os resultados das estratégias de tratamento de drift? Foi observado que estratégias de tratamento de drift são relevantes. Seu impacto varia de acordo com a escala utilizada. A avaliação experimental foi realizada usando um dataset que integra dados de clima e voos do sistema aéreo do Brasil. Além disso, as estratégias passiva e ativa mostraram melhores escores de recall. Em relação ao $f1$, as estratégias tiveram resultados similares, com a estratégia passiva mostrando resultados ligeiramente melhores. Esses resultados podem estar relacionados com a alta incidência de drifts. Nesse caso, estratégias que sempre retreinam modelos de aprendizado de máquina oferecem melhores resultados que aqueles que treinam somente uma vez. Entretanto, testes extensivos são recomendados.

Keywords: Flight delays, concept drift, machine learning, classification

Rio de Janeiro,

October 2021

ABSTRACT

Analyzing Flight Delay Prediction Under Concept Drift

Lucas Giusti Tavares

Advisors:
Jorge de Abreu Soares e Eduardo Soares Ogasawara (Co-advisor)

Abstract of dissertation submitted to Programa de Pós-graduação em Ciência da Computação - Centro Federal de Educação Tecnológica Celso Suckow da Fonseca CEFET/RJ as partial fulfillment of the requirements for the degree of master.

Delay is one of the most critical indicators for flight transportation systems. Flight delays impose a challenge that impacts any flight transportation system. In this context, the prediction of delayed flights may be an essential tool for effectively addressing this problem. This dissertation investigates the prediction performance of different drift handling strategies in aviation under different scales. It considers two different scales: *system-based* (SB) and *airport-based* (AB). In (SB), all airports in the flight system are considered together. Conversely, in AB, each airport is studied separately. Specifically, this work proposed and answered two research questions: (i) How do drift handling strategies influence the prediction performance of delays?; and (ii) Do different scales change the results of drift handling strategies? It was observed that drift handling strategies are relevant. Their impact varies according to the scales used. The experimental evaluation was done using a dataset that integrates weather and flight data from the Brazilian system. Moreover, the *passive* and *active* strategies revealed better recall scores. For $f1$ scores, the strategies had similar results, with the passive strategy showing slightly better results. These results may be related to the high prevalence of drifts. In this case, strategies that always retrain machine learning models offer better results than those that train only once. However, extensive testing is recommended.

Key-words: Flight delays, concept drift, machine learning, classification

Rio de Janeiro,

October 2021

Contents

List of Figures

List of Tables

List of Abbreviations

Chapter I Introduction

Delay is one of the most critical indicators for flight transportation systems. Flight delays impose a challenge that impacts any flight transportation system. In the United States (US), it is estimated that a 10% decrease in flight delays would mean a US$ 8 billion (year base 2013) increase in Gross Domestic Product (GDP) Peterson et al. [2013]. In this context, the prediction of delayed flights may be an essential tool for effectively addressing this problem.

The development of machine learning models makes it possible to identify potentially delayed flights or critical periods before happening, enabling better planning. For that reason, many predictive models have been developed to achieve the task Rong et al. [2015]; Kim et al. [2016]; Yu et al. [2019]. Commonly, flight data is combined with weather information from departure and arrival locations to help predict flight delays Du et al. [2018]; Wu and Law [2019].

From the machine learning point of view, predicting delay may be a regression or classification task. In the former, the goal is to predict the time (usually measured in minutes) that a flight will delay. In the latter, the goal is to predict whether the flight is going to delay Kim et al. [2016]; Gui et al. [2020a]. The literature specialized in flight delay prediction provides many different models that have been developed with good results for both tasks. Specifically, for classification, which is the focus of the present study, random forest and deep recurrent neural network models have shown promising results in the US and China datasets Rong et al. [2015]; Yu et al. [2019].

Large flight systems (such as the US, China, Europe, and Brazil) have challenges that impact flight delays. The relation of delays with input variables, such as destination or weather, may vary according to time and space Sternberg et al. [2016]. Thus, the proportion of delays may vary from time to time. Such a variation may occur due to punctual events such as storms and strikes. Other variations are disruptive, such as the FIFA World Cup in 2014. It led to an increase in the airports' capacity throughout most of the Brazilian Flight System and a significant flight change.

Nevertheless, even when there is no perceived change in the size of the flight system, the distributions of each feature and the relationship between system features and delays may vary. These relationship variations lead to a scenario called *concept drift* Iwashita and Papa [2019]. When the variables change but do not interfere with how delay occurs, there is no concept drift. Generally, a concept drift is a (statistically significant) difference between the joint probability of input and output variables observed in different dataset samples.

Previous studies indicate that concept drift may impact predictive models Gama et al. [2014]; Webb et al. [2016]; Iwashita and Papa [2019]. Some studies have tested algorithms that retrain the aviation models if drift is detected or used algorithms that may adapt to concept drift, like recurrent neural networks Khamassi and Sayed-Mouchaweh [2014]; Pesaranghader and Viktor [2016]; Kim et al. [2016]. Moreover, the amount of data used to train each model (so-called *scale*) varies across past studies: models trained from flights related to a single airport or the entire flight system. To the best of our knowledge, no study considered different system scales.

This dissertation investigates the prediction performance of active and passive drift handling strategies in aviation under different scales. It considers two different scales: *system-based* (SB) and *airport-based* (AB). In SB, all airports in the flight system are considered together. Conversely, in AB, each airport is taken into account separately. Specifically, two research questions were proposed and answered: (i) How do drift handling strategies influence the prediction performance of delays? (ii) Do different scales change the results of drift handling strategies? This dissertation proposed Stealthy, a framework to evaluate different drift detection methods and drift handling strategies under different time windows and scales to answer these questions. For that, we also used the Brazilian flight system dataset. It is an integrated database containing flight operations data provided by the Brazilian National Civil Aviation Agency (ANAC) ANAC [2017] and airport weather data provided by Automated Surface Observing Systems (ASOS) ASOS [2019]. The main contributions to the state-of-the-art of the present work were the following:

- Always retraining the models may offer better recall with similar precision scores.
- Training one model for each airport show generally better predictive performance;
- Training with higher periods (1, 2, and 3 years) showed good results, but windows longer than one year are not significantly better for prediction performance;
- Random Forests and Neural Networks were significantly better than Naive Bayes;
- Important performance indicators like precision, recall, and $f1$ were relevant to understand prediction performance. However, other studies did not use these indicators.
- Extensive testing is recommended;

Besides this introduction, this dissertation is organized as follows. Chapter II presents the general background for delay prediction and concept drift. Chapter III presents the related work. Chapter IV discusses the methodology (Stealthy Models) used for drift analysis over SB and AB. Chapter V presents experimental evaluation. Finally, Chapter VI presents some concluding remarks and points out future work.

Chapter II Background

The background is divided into two parts. Chapter II.0.1 presents flight delays prediction using machine learning. Chapter II.0.2 presents concept drift, including drift detection and handling.

II.0.1 Flight Delays Prediction Using Machine Learning

Flight delay is a measure of the actual departure or arrival time minus their respective expected time. For the classification task of predicting whether a delay will occur or not, a threshold is set up to establish this binary variable. Most studies use a threshold of 15 minutes. It indicates that any flight delayed for 15 minutes or more is marked as delayed Gui et al. [2020a]; Guleria et al. [2019a]; Kim et al. [2016]; Sternberg et al. [2016].

Many studies have been conducted and reported good results in the classification task Belcastro et al. [2016]; Moreira et al. [2018a]. In this context, Random Forests (RF) and Neural Networks (NN) achieved better *accuracy*. An NN is a bio-inspired computational approach that performs the processing of information through neurons that are connected through synapses Han et al. [2011]. Specifically, Multi-Layer Perceptrons associated with Long Short Term Memory Recurrent Neural Network (LSTM-RNN) and Deep Multilayer Perceptrons seem to show good results during prediction Gui et al. [2020a]; Kim et al. [2016]; Rebollo and Balakrishnan [2014]. Due to its interpretability, Naive Bayes (NB) is commonly included in the studies, as it encompasses a Baseline method. NB is a statistical classifier that can predict the probability of a tuple belonging to a particular class.

In a traditional classification problem, a dataset is separated into training and test sets. The model is built using the training set. It is common to partition the data set using a cross-validation strategy to optimize hyperparameters. Once the model is built, it is later evaluated using a test set Han et al. [2011]. This traditional approach is depicted in Figure II.1.a.

However, for flight delay prediction, the time dimension is relevant. Flights occur continuously each day as a streaming data source. Even when the entire streaming is stored in a single dataset, the time stamp of the flight events needs to be considered. It means that training should occur using past data to predict more recent data. It is depicted in Figure II.1.b, where the i-th batch (training data) is used to build a model for further evaluation with more recent data at the next batch $(i + 1)$ (test data). It is worth mentioning that the i-th batch corresponds to a sample of the dataset in the time interval associated with i. While studying concept drift, such methodology is mandatory Iwashita and Papa [2019].

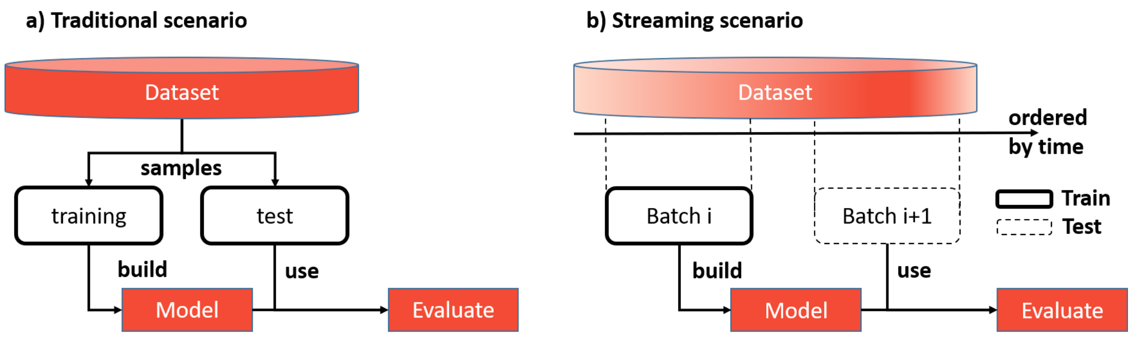


Figure II.1: Characterization of training and testing during classification: (a) traditional scenario; (b): streaming scenario

Finally, metrics are used to evaluate the prediction performance of built models. Formally, two classes (positive and negative) were given: positive tuples corresponding to delays and negative tuples for those without it. P is the number of positive tuples, and N is the number of negative tuples. The classes of the test set are compared to the classes predicted by the built model, getting: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) Han et al. [2011]; Moreira et al. [2018a]. From these measures, it is possible to compute the most widely used metrics: $accuracy$ ($\frac{TP+TN}{P+N}$), $precision$ ($\frac{TP}{TP+FP}$), $recall$ ($\frac{TP}{TP+FN}$), and f_1 ($\frac{2 \times precision \times recall}{precision+recall}$).

II.0.2 Concept Drift

Consider a classification problem, such that a set of input variables X is used to predict a class label Y. One of the main challenges when creating machine learning models is handling concept drift. It refers to a significant change in data distribution that interferes with the relation between the output class Y and input variables X. Formally, a concept at a time i is defined as the probability of the joint distribution χ of X and Y. It is described in Equation II.1. A concept drift between time i and j is defined as a difference (with statistical significance) between the probabilities $p_i(\chi)$ and $p_j(\chi)$. It is described in Equation II.2 Webb et al. [2016].

$$concept_i = p_i(\chi) = p_i(X, Y) \tag{II.1}$$

$$p_i(\chi) \neq p_j(\chi) \tag{II.2}$$

The drifts can also be classified as real or virtual drifts Iwashita and Papa [2019]. Specifically, real concept drifts are defined by changes in the posterior probabilities $p(Y|X)$, commonly related to the class boundaries. Conversely, virtual concept drifts happen whenever the conditional probability

$p(X)$ changes Iwashita and Papa [2019]; Lu et al. [2018]; Hoens et al. [2012]. A visual example is shown in Figure II.2.

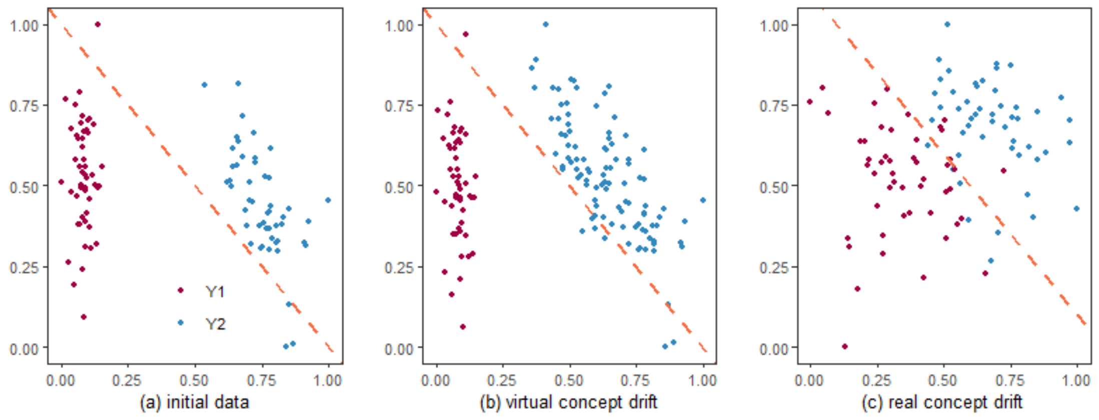


Figure II.2: Examples of types of drifts: (a) initial dataset; (b) virtual concept drift from (a); (c) real concept drift from (a)

Regarding flight delays, there can be relevant changes in the proportion of delayed flights. Some periods are more critical than others Gui et al. [2020a]; Sternberg et al. [2016]. For that reason, detecting and handling drifts is a relevant subject. Drift detection refers to the task of identifying concept drifts. It enables a specific action to avoid increasing errors in online learning systems after drift is observed Iwashita and Papa [2019]; Lu et al. [2018]; Webb et al. [2016]. There are two main categories of drift detection Lu et al. [2018]: (i) data distribution and (ii) error rate. Data distribution-based methods use statistical inference and analysis of feature distribution to detect significant output class proportion changes concerning its input variables. Error rate methods use machine learning algorithms and indicate a drift based on the error rate of prediction results. The detection of drifts can be based solely on data distribution, the error rate of predictions, or both.

Consider a dataset or a streaming dataset (D) partitioned into batches (time intervals of the same size). D_1 and D_n correspond to the first and last batches of D, respectively. Yet, a batch sequence b at time i is formally defined as $seq_{i,b}(D) = <D_{i-b+1}, \cdots, D_i>$. Indeed, a Batch Sequence Size (BSS) equals b establishes a sliding window to explore all batch sequences of size b present in D. It can be used to target both the detection and handling of concept drifts. It can be formalized as $sw_b(D)$. It corresponds to a matrix W of size $(n - b + 1)$ by b. Each line w_i in W is the i-th BSS b in D. Given $W = sw_b(D)$, $\forall w_i \in W$, $w_i = seq_{i,b}(D)$ Iwashita and Papa [2019]; Lu et al. [2018].

From these concepts, it is possible to define three strategies to address concept drift: (i) *baseline*; (ii) *passive*; (iii) *active*. In the *baseline* strategy, a model is built using the first batch. The trained model is continuously used. When drift occurs, no action is done, and the trained model might increase its error when predicting newer batches. Considering BSS equals one ($b = 1$), it corresponds to Figure II.3.a, where the first batch (1) is used for training a model (indicated as a red square) for predicting all other batches (2 to n).

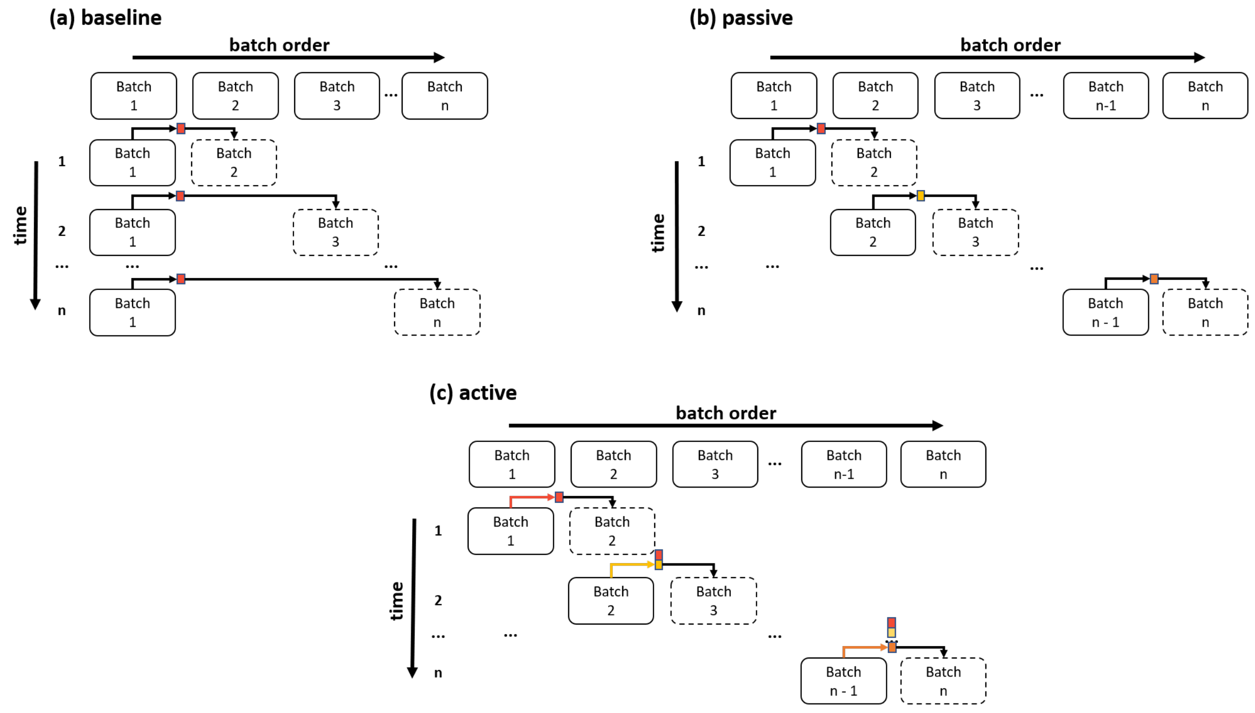


Figure II.3: Drift handling strategies considering a BSS equal to one ($b = 1$). (a): *baseline*; (b): *passive*; (c): *active*

In a *passive* strategy, it is assumed that drift occurs all the time. Thus, considering again a BSS equals one $b = 1$, batch i is used for training the model to predict batch $i + 1$. This scenario corresponds to Figure II.3.b. Models are constantly updated (they are presented in different colors). The drawback of this approach is that it might retrain models, even if no drift occurred in the dataset Iwashita and Papa [2019]; Gama et al. [2014].

Finally, in the *active* strategy, drift detection is applied whenever a new batch is introduced. If no drift is detected, the previously trained model is still used. However, if drift is detected, a new model is built using previous batches. Figure II.3.c depicts this scenario for $b = 1$. A new model (in orange) is used if drift occurs between batches two and three. Otherwise, the previous model (presented in red) is preserved. In this strategy, two extreme scenarios may occur. The same model can be used from the first batch to the last one, resembling the baseline strategy. The difference is that the decision is because no drift was observed. Conversely, continuous retraining may occur between each pair of batches, resembling the *passive* strategy. Again, such a decision is based on whether drift is observed whenever a new batch is introduced Iwashita and Papa [2019]; Gama et al. [2014].

Chapter III Literature Review

Two literature reviews were conducted to identify relevant data science studies related to 1) flight delay classification with the streaming approach; and 2) Brazilian database integration. For both cases, there were deemed only papers (Journals and Conferences) entirely written in English. The Scopus Database was selected due to having one of the most extensive and more accurate scientific bibliographic repositories [Cavacini, 2015]. The queries considered titles, keywords, and abstracts in February 2021.

III.1 Flight delay prediction with streaming approach

The review task aimed to identify relevant data science studies regarding flight delay classification with the streaming approach. After analyzing the main keywords related to the subject and synonyms, the final string used was: (”flight delay”) AND (“classification” OR “regression” OR “prediction”) and returned 141 papers, adding two studies through snowballing. The systematic review flow is shown in figure III.2.

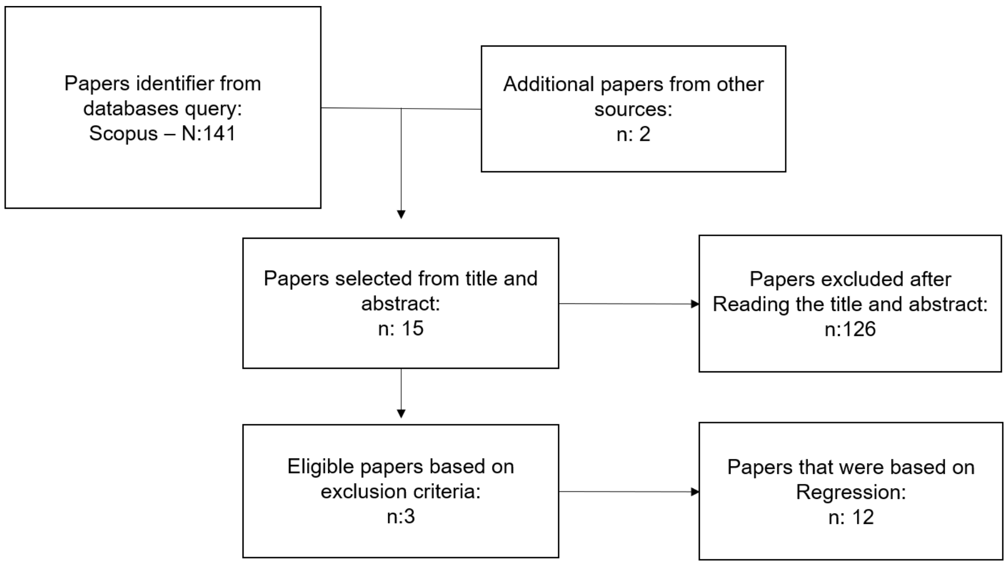

Papers identifier from databases query: Scopus – N:141
Additional papers from other sources: n: 2
Papers selected from title and abstract: n: 15
Papers excluded after Reading the title and abstract: n:126
Eligible papers based on exclusion criteria: n:3
Papers that were based on Regression: n: 12

Figure III.1: Flight delay prediction review flow

From the 141 articles found, only 33 were related to the main subject, included after reading titles and abstracts. Then, eight studies investigating only the regression task and 22 others that did not use the streaming approach were excluded. Finally, three articles investigated flight delay classification with a streaming approach described in the present work.

Regarding training and testing, the majority of studies uses traditional scenario (Figure III.2) [Chen et al., 2008; Khanmohammadi et al., 2014; Alonso and Loureiro, 2015]. However, this approach does not consider the possible drifts that are usually present on flight data. Studies that considered streaming approaches were deeply investigated, particularly those that contained drift handling strategies [Kim et al., 2016; Khamassi and Sayed-Mouchaweh, 2014; Pesaranghader and Viktor, 2016; Munoz Hernandez et al., 2019; Wang et al., 2019; Ai et al., 2019; Gui et al., 2020a]. In fact, only the work of Kim et al. [2016], Khamassi and Sayed-Mouchaweh [2014], Pesaranghader and Viktor [2016] investigated the binary classifier problem.

Kim et al. [2016] used Deep Neural Networks to predict flight delays in U.S. flights using a BSS of seven and nine days before prediction. An additional Deep Recurrent Neural Network was trained to classify the critical delay status of each day. Since the scale used was flight routes, these predictions were added to individual flight routes data and used as input in a Deep Neural Network. Although the *accuracy* was above 80% in most experiments, key metrics such as f_1, *precision*, and *recall* were not reported. As no specific strategy was used to identify or deal with drifts actively, we considered this approach a passive strategy.

Khamassi and Sayed-Mouchaweh [2014] proposed a new error active distance-based approach for drift detection and monitoring named EDIST. Specifically, EDIST compares new data with existing data and retrains the model if significant changes are found in the distribution. The classifier used in this case was the Hoeffding Trees, and BSS was statistically adaptive. Active strategies of the Drift Detection Method (DDM) and the Early DDM (EDDM) are both error-based methods. They were implemented for comparison reasons. A baseline strategy of training with the first batch and predicting the remaining data was tested as well. They have applied these techniques to many synthetic and real-world datasets (including the U.S. flight dataset). However, only the accuracy was used to report prediction performance.

Finally, Pesaranghader and Viktor [2016] tested many drift detection techniques with Naive Bayes and Hoeffding Trees. Specifically, their work proposes using the Hoeffding Inequality Theorem to test the difference in the probability of a given class between two BSS of 25 cases, characterizing an active strategy. DDM, EDDM, Adaptive Sliding Window (ADWIN), Hoeffding Drift Detection Method (HDDM), and Fast HDDM (FHDDM) active strategies were implemented for comparison reasons. ADWIN implements statistically adaptive batch sizes, defined whenever a drift is detected by average comparison. HDDM uses the Hoeffding inequality to compare distributions of batches. Finally, the proposed FHDDM uses Hoeffding inequality to compare errors from batches and thus detect drifts. The reported *accuracy* was around 65% for all experiments with aviation data. Fast Hoeffding Trees with Adaptive Windowing showed the best results for the U.S.

flight datasets.

To the best of our knowledge and considering the systematic review presented here, no study compared active and passive drift handling strategies for delay prediction in aviation. Moreover, the influence of the scale of the data used to train the models was never investigated as an essential factor for delay prediction. Besides accuracy, no other prediction performance indicators were reported as well.

III.2 Flight Database Integration

A second literature review was conducted to identify relevant data science studies regarding flight database integration. The main keywords and their synonyms were analyzed, and the final string used was: (flight) AND (database) AND (weather) AND (integration) AND NOT (bird) AND NOT (mammal) AND NOT (DNA) AND NOT (ran) AND NOT (urine) AND NOT (fertilizer) AND NOT (helicopter) AND (LIMIT-TO (DOCTYPE, "ar") OR LIMIT-TO (DOCTYPE, "cp")), returning 420 papers.

From the 420 articles found, 119 were related to commercial flight subjects and were included after reading the title and abstract. From those, studies with flights trajectories datasets were excluded. Only three articles used integrated datasets with commercial flights and weather data from the U.S., China, India, SE Asia, Germany, and Brazil. Additionally, 28 papers were added from the first string results, resulting in 31 studies. Moreover, only one study from these results used an integrated dataset from Brazil.

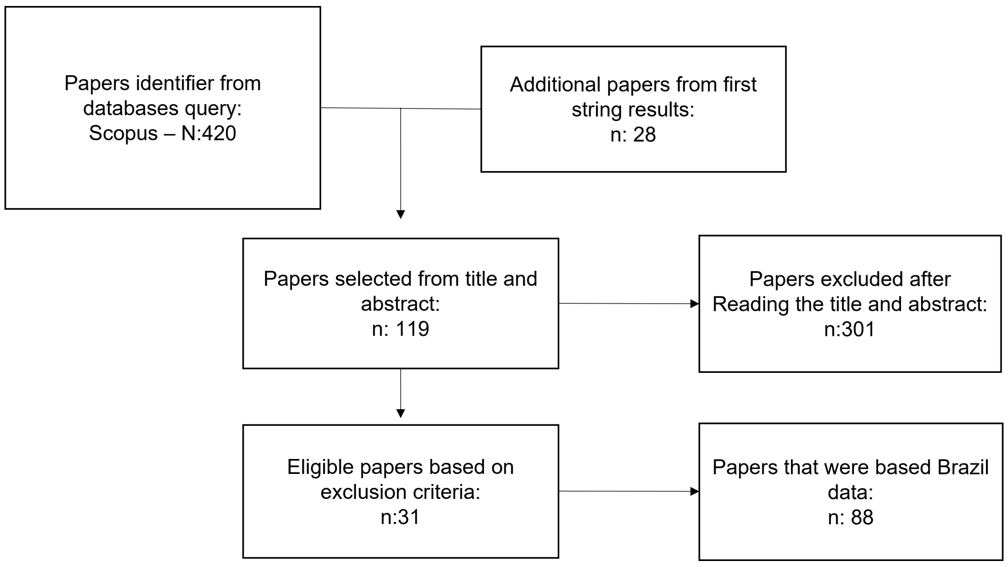

Papers identifier from databases query: Scopus – N:420
Additional papers from first string results: n: 28
Papers selected from title and abstract: n: 119
Papers excluded after Reading the title and abstract: n:301
Eligible papers based on exclusion criteria: n:31
Papers that were based Brazil data: n: 88

Figure III.2: Flight Database Integration review flow

Regarding countries, U.S. was the most frequent, with 21 articles using integrated flights datasets from there. However, all U.S. studies used the same sources to integrate the datasets used. From those 32 selected articles, the leading sources were described in the present work. A summary of the final selected articles is presented in Table III.1.

The U.S. Department of Transportation was the most common source of data for the articles that used the U.S. flights dataset with 16 citations. Specifically, regarding the other two, one was a public dataset from Kaggle [Yanying et al., 2019] and the other did not state the source [Nigam and Govinda, 2018]. Moreover, five articles integrated weather data to original flight data. From those, four were from National Oceanic and Atmospheric Administration (NOAA) [Chen and Li, 2019; Choi et al., 2016, 2017; Kalyani et al., 2020; Chandramouleeswaran et al., 2018] and one from World Weather Online (WWO) API [Thiagarajan et al., 2017].

Other countries studied were China with six articles, Brazil with two articles, and one article to India, Iran, and Germany. One study used data from Southeast Asia that investigated Indonesia, Vietnam, Cambodia, Malaysia, Singapore, Laos, Myanmar, Thailand, Philippines, Brunie, and Timor-Leste. Specifically, regarding China articles, the sources were the airports or online websites. Only two of those used weather data from the same source [Gui et al., 2020b; Liu et al., 2020]. The two articles that investigated the Brazilian air system used the Civil National Aviation Agency (ANAC) [Arnaldo Scarpel and Pelicioni, 2018] for flight data. One used weather data from NOAA [Moreira et al., 2018b] Moreover, Iran flight data source was a non-identified airline and the Iran Meteorological Organization for weather [Khaksar and Sheikholeslami, 2019]. Germany's was the Frankfurt Airport for flights and weather [Rehm and Klawonn, 2005]. The article that investigated 11 countries from Southeast Asia used Automatic Dependent Surveillance-Broadcast (ADS-B) for flight information [Guleria et al., 2019b].

Finally, it was clear that the primary sources for the flight came from government institutions that regulate air traffic. Moreover, NOAA weather data was the most frequent source for this kind of data. To the best of our knowledge and considering the literature review presented here, only two studies used an integrated dataset with commercial flight and weather data from Brazil [Moreira et al., 2018b; Sternberg et al., 2016].

Table III.1: Summary table of selected studies.

Author	**Year**	**Country**	**Source**	**Weather**
Anderson et al	2019	US	FAA	No
Baluch et al	2017	US	BTS	No
Chen e Li	2019	US	BTS	NOAA
Choi et al	2016	US	BTS	NOAA
Choi et al	2017	US	BTS	NOAA
Ganesan et al	2010	US	ASPM	No
Kalyani et al	2020	US	BTS	NOAA
Manna et al	2017	US	BTS	No
Meel et al	2020	US	BTS	No
Natarajan et al	2018	US	BTS	No
Nigam e Govinda	2017	US	N/A	N/A
Saadat e Moniruzzaman	2019	US	BTS	No
Tan et al	2018	US	ASA	No
Teja et al	2019	US	BTS	No
Thiagarajan et al	2017	US	BTS	WWO
Yanying et al	2019	US	Kaggle	No
Chandramouleeswaran e Krzemien	2018	US	BTS	NOAA
Fleurquin et al	2013	US	BTS	No
Sridhar et al	2009	US	BTS	No
Chen e Yan	2008	China	N/A	N/A
Ding	2017	China	Umetrip.com	No
Gu et al	2020	China	Shenzhen Airport	No
Gui et al	2020	China	Tianqihoubau.com	Ctrip.com
Liu et al	2020	China	Tianqihoubau.com	Ctrip.com
Yu et al	2019	China	Beijing PEK Airport	No
Haripriya e Ramyasree	2020	India	N/A	N/A
Khaksar e Sheikholeslami	2019	Iran	Iran Airline	IRIMO
Rehm e Klawonn	2005	Germany	Frankfurt Airport	Frankfurt Airport
Scarpel e Pelicioni	2018	Brazil	ANAC	No
Moreira et al	2018	Brazil	ANAC	NOAA
Guleria et al	2019	SE Asia	ADS-B	No

Chapter IV Methodology

This dissertation aims to study drift handling strategies. Specifically, three different *active* strategies and one *passive* strategy were investigated. The selected techniques showed promising results in past studies but were never evaluated with flight data sets. Therefore, our analysis was based on this comparison (research question one). A second goal is to investigate drift handling strategies under the influence of different scales of training data: System-Based (SB) and Airport-Based (AB). In other words, how the scales interfere with drift handling strategies (research question two). It is worth mentioning that, to the best of our knowledge, the scale of training data was never analyzed in previous studies.

In this chapter, the general methodology (named Stealthy Models) used in the present work is described (Chapter **??**). It comprises three steps: 1) Preprocessing; 2) Drift Detection and Handling; 3) Model training and evaluation. Step 1 (Chapter IV.0.2) explains the preprocessing methods implemented. Step 2 (Chapter IV.0.3) consists of all drift-related strategies and techniques implemented. Finally, step 3 (Chapter IV.0.4) shows how the models are trained.The following subsections explain the general principle of Stealth Models and each step.

IV.0.1 Stealthy Models

Algorithm 1 describes the methodology used for flight delay prediction with concept drift. It requires seven parameters: D, $airport$, mlm, t, b, dd, and dh. The parameter D corresponds to the input dataset. The parameter $airport$ identifies a single airport if the AB scale is used, or nil if the SB is used. Parameter mlm corresponds to the machine learning method. Parameter t is related to the time, and b corresponds to the size of the batch sequence. Finally, dd and dh correspond to the drift detection method and drift handling strategy, respectively. These parameters are described in Table IV.1.

The first step of Stealthy 1 does the preprocessing (function *preprocess*). If *airport* is different from nil, the dataset D is filtered for that airport. Otherwise, it studies the entire SB. All steps of data preprocessing are described in Chapter IV.0.2. Line 4 implements a loop through all batches available within the dataset time frame available.

Lines 5 and 6 computes the drift detection and handling. It considers the drift detection method (dd), drift handling strategy (dh), and the two batch sequences (D_i and D_j). The mechanics of drift

Algorithm 1 General methodology

```
1: function $Stealthy(D, airport, mlm, t, b, dd, dh)$
2:     $D_p \leftarrow preprocess(D)$
3:     $results \leftarrow \emptyset$
4:     for each $i \in t$ do
5:         $train, D_i \leftarrow checkDrift(D_i, airport, t, b, dd, dh)$
6:         $mdl \leftarrow cloakModel(train, mlm, D_j)$
7:         $results \leftarrow results \cup predictEvalDelays(mdl, D, t)$
8:     end for
9:     return $results$
10: end function
```

Table IV.1: Parameters used

Name	**Description [values]**
D	Dataset with flight and weather data
airport	Airport code for AB analysis; or *nil*, for SB analysis
mlm	Machine learning models
t	Time (yearly based time slices)
b	BSS for training
dd	Drift detection method
dh	Drift handling strategy

action are described in Chapter IV.0.3. The output of drift action is assigned to *train*, indicating if training is required at time t. If required, a machine learning method is trained using the batch sequence D_i (Line 6). The training process is described in Chapter IV.0.4. The output is a trained model (*mdl*). The model is stored for the selected *airport*, *mlm*, *dd*, and *dh*. If no training is required, previously trained model *mdl* is retrieved for the selected *airport*, *mlm*, *dd*, and *dh*. Line 7 adds the results from *predictEvalDelays* in a dataset. After batches are processed, the dataset with the results is returned.

IV.0.2 Preprocessing

In our proposed workflow, the first step is to preprocess the data. The function *preprocess* expects a dataset with a binary delay feature and a DateTime feature to calculate the weekly proportion of delays used for drift checking. Then, the week number within the year is created to group and aggregate by the proportion of delayed flights. Statistical tests for drift detection are performed using the weekly values of delay proportion.

Regarding data cleaning, the *preprocessing* function removes cases with negative or too long durations (more than 19h) as they are considered errors, considering this flight duration in BFS is not possible. Cases with missing delay values or estimated departure dates are also removed. For other features, mode imputation is used with categorical ones and mean for numeric data.

Finally, the preprocessing function returns a full preprocessed dataset with a binary delay feature. The new feature of the week number is also returned with the dataset. If an airport is passed, the dataset is reduced to the specific flights that departed from that airport. If *nil* is passed as airport, the entire dataset with all airports is returned.

IV.0.3 Drift Detection and Handling

Regarding drift detection and handling, three methods and three strategies for drift detection and handling, respectively, were implemented. The drift handling strategies were the *baseline*, *active*, and *passive* drift handling strategies (Figure II.3). The *baseline* corresponds to training using the first batch sequence to predict all other batches. The *passive* strategy trained a model for each batch sequence to predict the next batch. Finally, the *active* strategy compared the current training batch sequence with the previous one to detect a drift. In case of not having drift, the previously trained model is chosen. Otherwise, a new model is trained using the current training batch sequence. The prediction of the next batch is made using the chosen model. All drift handling strategies were tested for the SB and AB scales. The function that implements all drift features and returns whether a model should be trained and the corresponding training data is described in Algorithm 2.

For *active* strategies, we implemented three methods of drift detection based on data distribution analysis. They evaluate the occurrence of drifts according to the proportion of delays ($p(Y)$) presented in two consecutive batch sequences (D_i and D_{i-1}). The three methods were (i) *mean*, (ii) *variance*, and (iii) *mean/variance*. In *mean* and *variance*, respectively, the *mean* and *variance* of both training batches sequences were compared for a statistically significant difference. Finally, in *mean/variance*, a significant difference in either *mean* or *variance* indicates a drift. Normality was tested with Kolmogorov Smirnov, and Shapiro-Wilk tests Yap and Sim [2011]. Most commonly used tests for comparison of means and variance were used. Specifically, the mean t-test was used for normal distributions, and the F test was used for the *variance*. When distributions were not normal, the Wilcoxon test was used for the *mean*, and the Levene test was used for the *variance*. The p-value used was 0.05 for all tests.

The function $checkDrift$ (Algorithm 2), shows how drift handling and detection are implemented. First, the two most recent batches considering the t parameter are created from the full dataset D. Then, the drift detection (dd) and drift handling (dh) parameters are used to check the presence of drifts between the two most recent batches. Finally, if drift is detected, the *train* variable is returned True or False otherwise. The variable D_i is also returned to be used in $trainModel$ function.

Algorithm 2 Drift Detection Function

```
1: function $checkDrift(D, airport, t, b, dd, dh)$
2:     $D \leftarrow preprocess(D, airport)$
3:     $D_i \leftarrow selectBatchesTrain(D, t, b)$
4:     $D_j \leftarrow selectBatchesTrain(D, t-1, b)$
5:     $train \leftarrow actDrift(dd, dh, D_i, D_j)$
6:     return $train, D_i$
7: end function
```

IV.0.4 Model training and evaluation

In our proposed approach, a model is set for each step $i \in t$. Algorithm 3 was used for model training/loading. The function *cloakModel* receives the parameters from the $checkDrift$ function as well as the airport (*airport*), machine learning model (*mlm*), time (t), BSS (b), drift detection method (*dd*) and drift handling strategy (*dh*). $cloakModel$ is a straightforward function that trains a new model if *train* is true, or loads the last model otherwise. The $trainModel$ function also implements a grid search with a 10% random sample of the batch data used for training to optimize models hyperparameters.

Algorithm 3 Model Setting Function

```
1: function $cloakModel(train, D, airport, mlm, t, b, dd, dh)$
2:     if $train$ then
3:         $mdl \leftarrow trainModel(mlm, D_i)$
4:         $storeModel(mdl, airport, mlm, dd, dh)$
5:     else
6:         $mdl \leftarrow loadModel(airport, mlm, dd, dh)$
7:     end if
8:     return $mdl$
9: end function
```

After model training, the predictions are made and prediction performance is evaluated with a specific function *predictEvalDelays* (Algorithm 4). This function predicts the next batch ($t + 1$) and evaluate the prediction performance. Four performance metrics described at Chapter **??** were implemented: 1) *accuracy*; 2) *precision*; 3) *recall*; 4) $f1$.

Algorithm 4 Evaluation Function

```
1: function $predictEvalDelays(mdl, D, t)$
2:     $D_{t+1} \leftarrow selectBatchTest(D, t+1)$
3:     $results \leftarrow test(mdl, D_{t+1})$
4:     return $results$
5: end function
```

Chapter V Experimental Evaluation

Considering the two main research questions proposed, chapters V.0.2, V.0.3 and V.0.4 are driven to answer how do drift handling strategies influence the prediction performance of delays. Chapter V.0.5 answers if different scales change the results of drift handling strategies. Finally, Chapter V.0.6 shows the results of the Time Efficiency analysis.

V.0.1 Experimental Setup

Considering the general methodology described in Algorithm 1 and parameters described in Table IV.1, Algorithm 1 describes the entire concept drift analysis. It executes the methodology considering the cross-product for all possible values of the parameters. The entire experimental evaluation was executed in one month on an i7 processor with 16 cores with 128GB of RAM and a Ubuntu 20.04 operating system.

The dataset used was the Brazilian Flights Dataset **?**. It is an integrated dataset containing ANACs flight operations ANAC [2017] with ASOS airport weather data ASOS [2019]. It contains data from 2000 to 2018. Three features were created. For destination, the state of each airport was used instead of the airport name. According to international standards for flight delay, the binary variable for flight delay was set with a 15 minutes limit. The cases where departure delay was higher than 19 hours or missing were considered errors and excluded from the analysis.

The batches used as test sets varied from 2004 to 2018 for all methods to compare different BSS for training (b from 1 to 3). The minimum value for BSS is one year to include yearly seasonality in each trained classifier, as we aim to create models that incorporate seasonal components of streaming data. The batches started in 2004, so all BSS may predict the same years, considering that the dataset started in 2000. This approach makes it possible to compare all combinations within the same period.

For the sake of scale comparison, the data used in this dissertation is filtered for the top ten airports with the highest number of departing flights, and only domestic flights were evaluated. Graphical representation of the location of airports is shown in Figure V.1, and descriptive information is shown in Table V.1. For the SB analysis, the filtered dataset is studied altogether. Conversely, for AB analysis, each one of the top ten airports is studied separately.

Regarding classifiers, four methods were used: NB, KNN, RF, and NN, as they were fre-

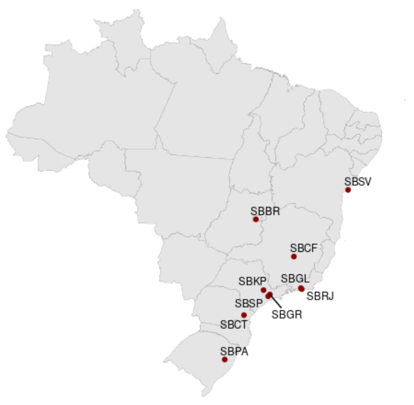


Figure V.1: Location of ten main Brazilian airports

Table V.1: Top ten Brazilian airports studied

Code	City	State
$SBBR$	Brasília	Distrito Federal (DF)
$SBSV$	Salvador	Bahia (BA)
$SBCT$	São José dos Pinhais	Paraná (PR)
$SBGL$	Rio de Janeiro	Rio de Janeiro (RJ)
$SBPA$	Porto Alegre	Rio Grande do Sul (RS)
$SBKP$	Campinas	São Paulo (SP)
$SBGR$	Guarulhos	São Paulo (SP)
$SBSP$	São Paulo	São Paulo (SP)
$SBCF$	Belo Horizonte	Minas Gerais (MG)
$SBRJ$	Rio de Janeiro	Rio de Janeiro (RJ)

quently used in other similar studies (Chapter **??**). Each technique was replicated ten times for each predicted year and BSS with 1/10 of the entire data to reduce the impact of outlier results from RF and NN, which are not deterministic and optimize execution time through parallelism. Moreover, the 9/10 for each execution was predicted and evaluated to cross-validate the models. For Naive Bayes tests, only one model was trained. A grid search was used to find the best hyperparameters for KNN, NN, and RF.

The hyperparameters were the number of hidden neurons for NN, the number of randomly selected predictors for RF, and the k parameter for KNN. Cross-validation was used to avoid overfitting, with the number of folds set to 10, which is the default value Schaffer [1993]. Moreover, to reduce complexity, the optimized hyperparameters were computed for all batches with 10% of the batch data.

The entire process was implemented in R for both preprocessing and machine learning methods Han et al. [2011]; James et al. [2013]. These are available as R packages (caret, nnet, randomForest, e1071, dplyr, PerformanceAnalysis).

Experimental evaluation

```
1: $A \leftarrow \{nil, SBBR, SBPA, SBSV, SBGL, SBCT, SBKP, SBGR, SBCF, SBRJ, SBSP\}$
2: $MLM \leftarrow \{NB, KNN, NN, RF\}$
3: $T \leftarrow \{2003, \cdots, 2018\}$
4: $B \leftarrow \{1, 2, 3\}$
5: $DD \leftarrow \{mean, variance, mean/variance\}$
6: $DH \leftarrow \{baseline, passive, active\}$
7: $results \leftarrow \emptyset$
8: for each $a \in A, mlm \in MLM, t \in T, b \in B, dd \in DD, dh \in DH$ do
9:     $results \leftarrow results \cup methodology(D, a, mlm, t, b, dd, dh)$
10: end for
11: return $results$
```

V.0.2 Comparison of Drift Detection Methods

The entire experimental evaluation execution considered the period from 2004 to 2018. In a yearly-based analysis, the total number of possible drifts is 15. The number of drifts is shown in Table V.2 for SB analysis. They are presented according to BSS.

Table V.2: Number of detected drifts for SB by BSS

BSS (training:test)	$mean$	$variance$	$mean/variance$
1:1	9	2	10
2:1	11	5	11
3:1	8	4	8

Considering $mean$ and $mean/variance$ methods, the average number of drifts over the entire period was 9.33 and 9.66, respectively. These numbers correspond to 62.2% and 64% of all possible drifts (15), which indicates a high drift prevalence between batches for all BSS sizes. Moreover, the $variance$ method showed an average of 3.66 (24%). Overall, the results show a high number of drifts in the dataset. Therefore, most drifts may be related to the increase or decrease of delays proportions between weeks, but not with the variance changes of this proportion.

V.0.3 Comparison of Drift Handling Strategies

A top-k analysis of the best combinations of parameters (drift handling strategy, drift detection method, classifier, and BSS) was also executed. First, the best combinations of parameters are ordered by $f1$ and divided into groups of drift handling/drift detection combinations. Then, for each group, the $accuracy$, $recall$, $precision$, $f1$ and $frequency$ are calculated for each top-k combinations of each group. The main results are summarized in Figure V.2.

Considering the accuracy (depicted in Figure V.2. A.), it is possible to see that Active strategies have higher or tied best scores more frequently than all the other strategies. Specifically, the Active strategy with drift detection based on mean seemed to have higher scores.

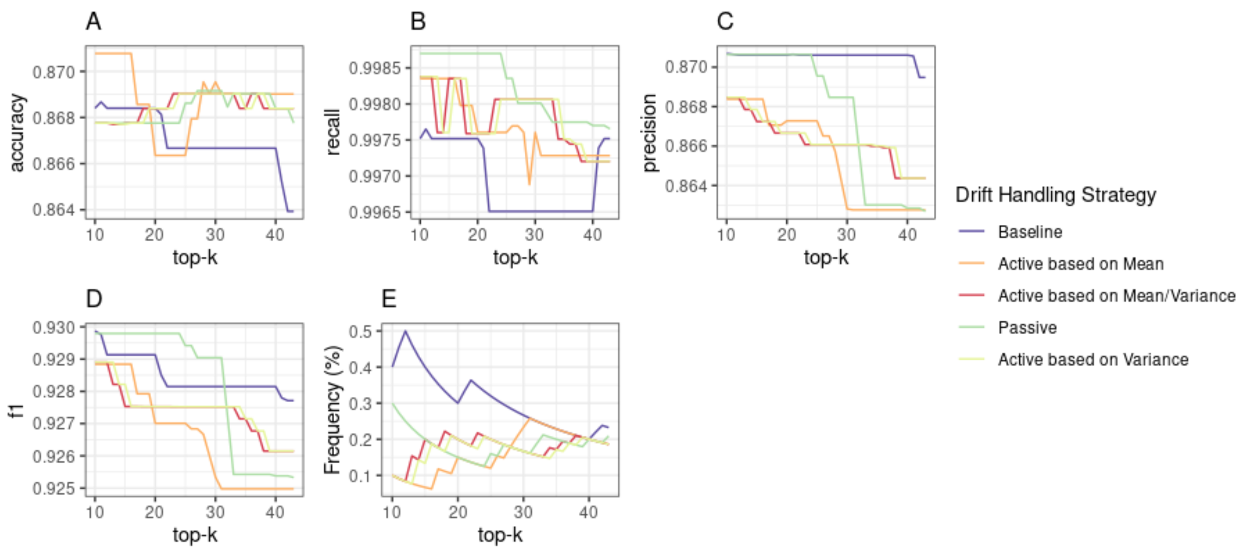


Figure V.2: Results of top-k analysis ranked by f_1. Mean performance for *accuracy* (A), *recall* (B), *precision* (C), f_1 (D). Frequency of each drift handling strategy (E).

Regarding the recall (depicted in Figure V.2. B.), Passive and Active strategies showed a higher performance for almost all k values used when compared to baseline strategy. Specifically, the Passive strategy showed the best performance throughout all k values. For Active strategies, all drift detection methods showed similar results. These results may indicate that new delay mechanisms may arise with time and, for that reason, only the strategies that have some adaptation to drifts may identify these cases, which may increase recall score.

Regarding the precision (depicted in Figure V.2. C.),Passive and baseline strategies showed higher precision scores until the $k = 25$. From there, passive precision decreased, and baseline showed higher scores. Regarding Active strategies, all drift detection methods had similar scores.

In $f1$ analysis (depicted in Figure V.2. D.), the passive strategy showed the higher values until $k = 31$. Active strategies showed lower values than baseline. These results may indicate that more retrains may be better for $f1$ performance for datasets with a high number of drifts. Specifically, considering recall and precision scores, the main difference may be in identifying new mechanisms of delayed flights, which may be possible with passive and active strategies.

The most frequent strategies among top-k were passive and baseline (depicted in Figure V.2. E.). Specifically, for the top-10, the difference was 10% and decreasing for higher values of k. These results may indicate that the baseline can offer high-scoring predictions more frequently. However, passive strategies may show higher performance scores. These results indicate that preserving baseline is essential to monitor the performance of active and passive strategies.

Although it was possible to think accuracy was good for active strategies, passive strategies showed good recall scores, and baseline strategies showed good precision. However, considering most past studies, if we analyzed only the accuracy, significant results about active and passive predictive performance would not be known.

V.0.4 BSS and Classifiers

Regression Modeling

To analyze the prediction performance of all parameters used in Algorithm 1, we conducted multiple linear regressions for each airport with the Drift Strategy-Detection combination, classifier, and BSS to estimate F1. After running all experiments, each result was stored as a row with columns for each category of each parameter. Specifically, each category of each categorical variable was considered a binary feature (classic one-hot encoding), and the Baseline Drift Handling Strategy, Naive Bayes Classifier, and one-year BSS were used as intercept (all zero columns). The final table used to train the linear regression had ten columns as shown in Table V.3.

Table V.3: Binary features used to train the linear regression model.

Feature	Parameter	Used as reference
Baseline	Drift Handling	*
Passive	Drift Handling	
Active based on Mean	Drift Handling	
Active based on Variance	Drift Handling	
Active based on Mean/Variance	Drift Handling	
BSS1	BSS	*
BSS 2	BSS	
BSS 3	BSS	
BSS 4	BSS	
Naive Bayes	Classifier	*
K-Nearest Neighbors	Classifier	
Random Forests	Classifier	
Neural Networks	Classifier	

With the one-year BSS used as a reference, positive or negative betas mean higher positive or negative impacts on $f1$ when compared to one year BSS. This interpretation may be used for Drift Handling and Parameters as well. Considering i independent variables (all parameters used to simulate the combinations), the linear regression would have $i + 1$ betas (one for each independent variable x) that are combined in a linear equation (Equation V.1) to estimate the dependent variable y (f in our case). These betas are optimized using a technique called least squares, indicating the amount of change in $f1$ when that value is used. The linear regression equations are shown below Groß [2012]. This analysis made it possible to understand the impact of each technique on $f1$ scores.

$$f1 = (\beta_0) + (\beta_1)x_1 + (\beta_2)x_2 + ...(\beta_i)x_i \tag{V.1}$$

Finally, each coefficient value is tested to check if it could be significant or by chance Groß

[2012]. Significant betas may indicate that the corresponding parameter has a significant impact over $f1$ compared to its reference option.

Comparison of BSS

The Betas results of linear regression models are shown in Figure V.3. Regarding BSS, the linear regression analysis showed that only SBSV, SBSP, SBRJ, and SBBR had significant betas for BSS greater than one. In those cases, all the values were negative, meaning that using a BSS higher than one would mean a lower $f1$ score. Specifically, the decreases caused by these BSS options would be around one percent in $f1$. These results may indicate that increasing the size may not be relevant to the $f1$ score, and the one-year size may be the best choice considering the values we tested. These results are shown in Figure V.3.

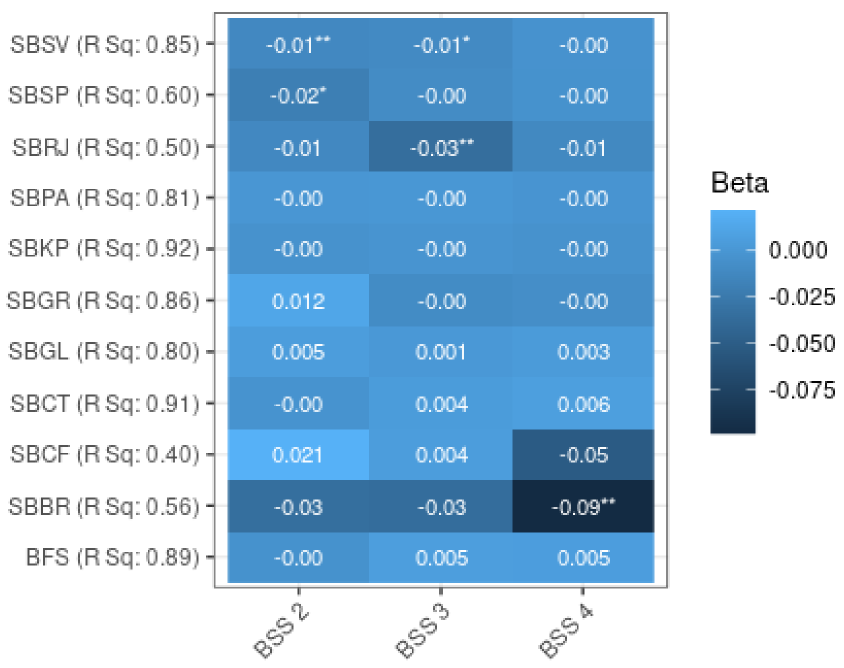
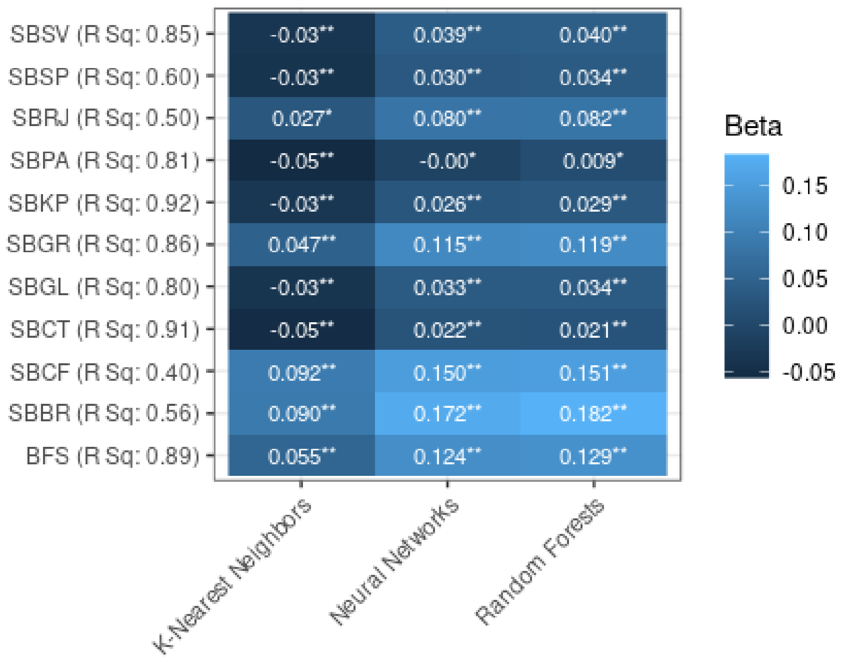


Figure V.3: Betas from Linear Regressions modeled for each BSS parameter used. $* = p < 0.05$. $** = P < 0.01$

Comparison of Classifiers

Using the same Regression analysis from Chapter V.0.4 to understand the influence of each classifier in $f1$, the NB classifier was used as a reference. Among classifiers, KNN, RF, and NN were always significant when compared to the NB reference. Specifically, the KNN showed similar results to reference NB, with small positive and negative Betas. NN and RF were almost always better for $f1$ than NB and KNN. The Beta for NN was negative only in SBPA airport, but all other cases showed significant positive values. Considering an RF/NN comparison, the difference in Betas was minimal, indicating that both may impact $f1$ in very similar ways.



	K-Nearest Neighbors	Neural Networks	Random Forests
SBSV (R Sq: 0.85)	-0.03**	0.039**	0.040**
SBSP (R Sq: 0.60)	-0.03**	0.030**	0.034**
SBRJ (R Sq: 0.50)	0.027*	0.080**	0.082**
SBPA (R Sq: 0.81)	-0.05**	-0.00*	0.009*
SBKP (R Sq: 0.92)	-0.03**	0.026**	0.029**
SBGR (R Sq: 0.86)	0.047**	0.115**	0.119**
SBGL (R Sq: 0.80)	-0.03**	0.033**	0.034**
SBCT (R Sq: 0.91)	-0.05**	0.022**	0.021**
SBCF (R Sq: 0.40)	0.092**	0.150**	0.151**
SBBR (R Sq: 0.56)	0.090**	0.172**	0.182**
BFS (R Sq: 0.89)	0.055**	0.124**	0.129**

Figure V.4: Betas from Linear Regressions modeled for each classifier parameter used. $* = p < 0.05$. $** = P < 0.01$

These results may indicate that the classifier used may be necessary when predicting flight delays in BFS. Specifically, more complex models (NN and RF) may have better results. The differences between NN and RF classifiers were minor, and it may not be possible, with the present results, to ascertain that one of them has better results.

Considering the other two classifiers (NB and KNN), the results were similar. Specifically, although the Betas from KNN were always significant, they were only positive with four airports (SBRJ, SBGR, SBCF, SBBR) and BFS. These results may indicate that between NB and KNN classifiers, it is not clear which of those could give better results.

V.0.5 Comparison of BS and AB approaches

As shown in Table V.4, the average number of drifts for the AB scale was very similar to SB. The most sensitive methods were also *mean* and *mean/variance*, with 8.6 (57% of all possible drifts) and 9.1 (61% of all possible drifts). The *variance* method was again less sensitive to drifts and showed a 4.4 average drifts (29.3% prevalence). These results indicate that the AB scale may not influence the number of drifts detected compared to the SB scale.

For a more in-depth analysis, Figure V.5 presents the metrics for all airports. Six airports showed better results than the BS approach. The AB approach seemed to improve prediction performance for SBRJ, SBCF, SBKP, SBPA, SBSV, and SBSP. Moreover, SBGL, SBCT, SBGR, and SBBR showed similar $f1$ results. SBBR was the only case with worse results than BFS. These results may indicate that performance improvements with the AB approach may be more likely to

Table V.4: Number of Drifts by each Airport, Drift Detection Method, and BSS.

Airport	*mean*			*variance*			*mean/variance*		
	1	2	3	1	2	3	1	2	3
$SBBR$	8	8	9	3	4	5	8	8	9
$SBCF$	9	10	9	2	4	5	9	10	9
$SBCT$	8	8	8	1	4	5	8	8	8
$SBGL$	8	10	9	2	4	6	8	10	9
$SBGR$	12	8	6	6	7	5	12	10	6
$SBKP$	11	11	10	5	8	4	12	13	10
$SBPA$	10	10	9	5	5	7	12	11	11
$SBRJ$	6	4	3	3	5	3	8	6	3
$SBSP$	9	7	7	3	3	3	11	7	7
$SBSV$	10	11	10	4	5	6	10	11	11
Mean	9.1± 1.7	8.7± 2.2	8.0± 2.2	3.4± 1.6	4.9± 1.5	4.9± 1.3	9.8± 1.8	9.4± 2.1	8.3± 2.5

occur and that a BFS model may be a good baseline for delay prediction.

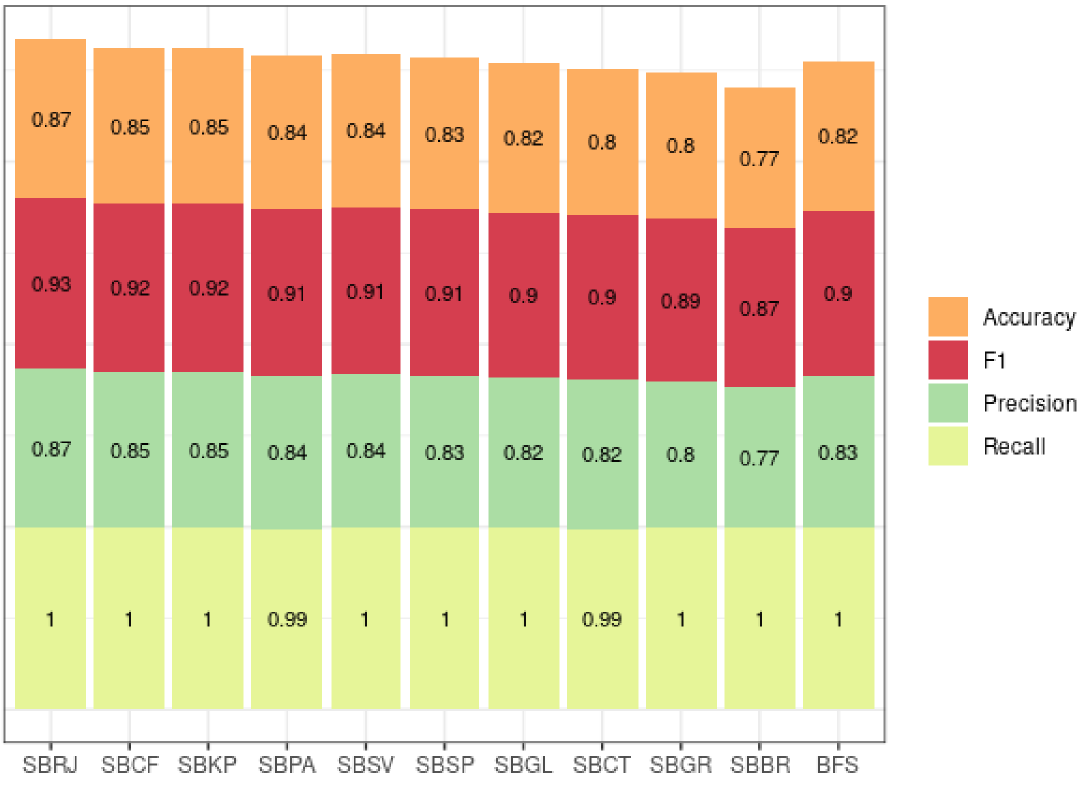


Figure V.5: Prediction performance metrics for airports.

Finally, by analyzing the data from Figures V.4, V.5 and Table V.2, it is possible to understand that, generally, the best scale of data to use may be the AB when it comes to model performance. However, the SB scale may help point toward airports needing more investigations in model training, being useful as a baseline model.

V.0.6 Time Efficiency Evaluation

To analyze the time efficiency, a new regression model was trained with the execution time as a dependent variable instead of $f1$. This approach made it possible to have similar insights about the importance of each technique used to time efficiency. Considering execution time, the baseline was the fastest option since it only trains once. Moreover, only four airports showed significance for drift strategy/detection method combinations. Specifically, the Passive was the category that increased the execution time the most. These results indicate that, although the Passive and Active strategies may train more than once, the overall increase in execution time was not significant compared to baseline for most airports.

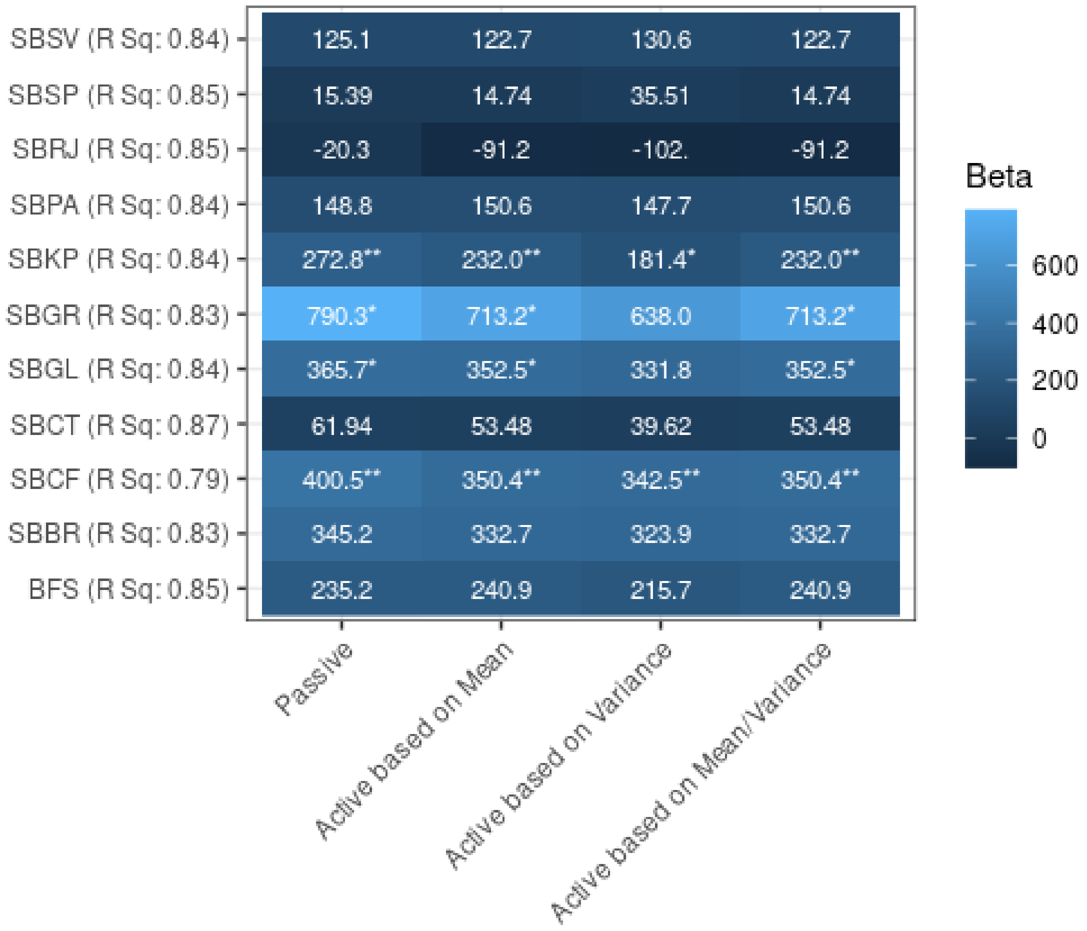


Figure V.6: Betas from Linear Regressions modeled for each Drift Handling parameter used. $* = p < 0.05$. $** = P < 0.01$

Considering Execution Time, the regression analysis show that the training time tends to be more significant as the BSS increases. This result is expected considering that the number of data points increases. However, considering that BSS higher than one does not significantly increase the prediction performance, an essential gain in execution time is possible.

Regarding execution performance, all models increased the execution time significantly when compared to NB. Specifically, NN was the slower option. Moreover, the Random Forests model shows betas of a quarter of NN's. These results may indicate that RF can be faster than NN, which is an essential game-changer since these two classifiers showed similar results in prediction performance.

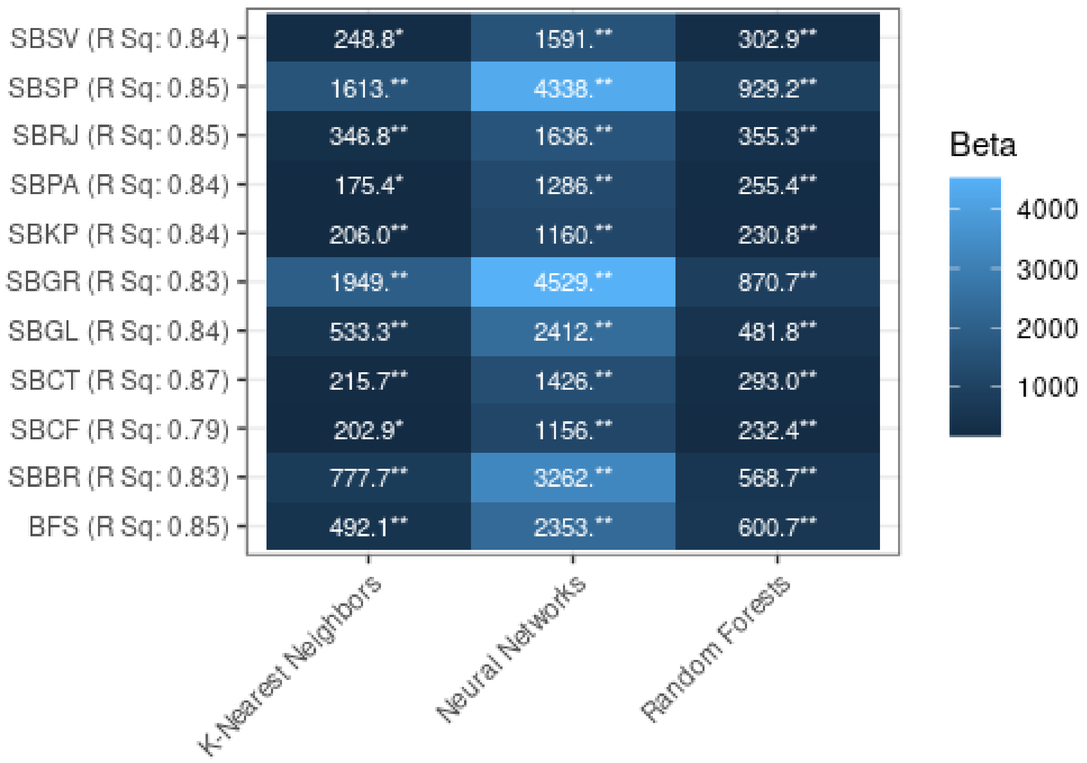


Figure V.7: Betas from Linear Regressions modeled for each classifier parameter used. $* = p < 0.05$. $** = P < 0.01$

Chapter VI Conclusions

In this paper, we analyzed different types of drift handling strategies in aviation. Two research questions were answered to achieve the main objective of this study: (i) How do drift handling strategies influence the prediction performance of delays? (ii) Do different scales change the results of drift handling strategies? We analyzed different strategies and scales and compared performances with linear regressions and a top-k analysis.

It was observed that drift handling strategies are relevant. Their impact varies according to the scales used. The experimental evaluation was done using a dataset that integrates weather and flight data from the Brazilian system. Moreover, the *passive* and *active* strategies showed better recall scores. For $f1$ scores, the strategies had similar results, with the Passive strategy showing slightly better results. It may be related to the high prevalence of drifts. In this case, strategies that always retrain machine learning models offer better results than those that train only once. However, extensive testing is recommended. Nonetheless, choosing machine learning models may have a higher impact on $f1$ than drift handling strategies.

As limitations, the first one is the drift detection methods used in the experimental evaluation. They were focused only on changes $P(Y)$. Future studies may consider testing error-based drift handling strategies, investigating ensemble drift detection methods, implementing more robust classifiers (CNN, LSTM, and others), check drift handling strategies under different thresholds for flight delays, and evaluate other metrics such as the ROC curve.

Bibliography

E.B. Peterson, K. Neels, N. Barczi, and T. Graham. The economic cost of airline flight delay. *Journal of Transport Economics and Policy*, 47(1):107–121, 2013. 1

F. Rong, L. Qianya, H. Bo, Z. Jing, and Y. Dongdong. The prediction of flight delays based the analysis of Random flight points. In *Chinese Control Conference, CCC*, volume 2015-September, pages 3992–3997, 2015. 1

Y.J. Kim, S. Choi, S. Briceno, and D. Mavris. A deep learning approach to flight delay prediction. In *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, volume 2016-December, 2016. 1, 2, 3, 8

B. Yu, Z. Guo, S. Asian, H. Wang, and G. Chen. Flight delay prediction for commercial air transport: A deep learning approach. *Transportation Research Part E: Logistics and Transportation Review*, 125:203–221, 2019. 1

W.-B. Du, M.-Y. Zhang, Y. Zhang, X.-B. Cao, and J. Zhang. Delay causality network in air transport systems. *Transportation Research Part E: Logistics and Transportation Review*, 118: 466–476, 2018. 1

C.-L. Wu and K. Law. Modelling the delay propagation effects of multiple resource connections in an airline network using a Bayesian network model. *Transportation Research Part E: Logistics and Transportation Review*, 122:62–77, 2019. 1

G. Gui, F. Liu, J. Sun, J. Yang, Z. Zhou, and D. Zhao. Flight delay prediction based on aviation big data and machine learning. *IEEE Transactions on Vehicular Technology*, 69(1):140–150, 2020a. 1, 3, 5, 8

A. Sternberg, D. Carvalho, L. Murta, J. Soares, and E. Ogasawara. An analysis of Brazilian flight delays based on frequent patterns. *Transportation Research Part E: Logistics and Transportation Review*, 95:282–298, 2016. 1, 3, 5, 10

A.S. Iwashita and J.P. Papa. An Overview on Concept Drift Learning. *IEEE Access*, 7:1532–1547, 2019. 1, 2, 4, 5, 6

J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4), 2014. 2, 6

G.I. Webb, R. Hyde, H. Cao, H.L. Nguyen, and F. Petitjean. Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4):964–994, 2016. 2, 4, 5

I. Khamassi and M. Sayed-Mouchaweh. Drift detection and monitoring in non-stationary environments. In *2014 IEEE Conference on Evolving and Adaptive Intelligent Systems, EAIS 2014 - Conference Proceedings*, 2014. 2, 8

A. Pesaranghader and H.L. Viktor. Fast hoeffding drift detection method for evolving data streams. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9852 LNAI:96–111, 2016. 2, 8

ANAC. The Brazilian National Civil Aviation Agency. Technical report, http://www.anac.gov.br/, 2017. 2, 16

ASOS. Automated Surface Observing Systems. Technical report, https://mesonet.agron.iastate.edu/request/download.phtml, 2019. 2, 16

Y. Guleria, Q. Cai, S. Alam, and L. Li. A Multi-Agent Approach for Reactionary Delay Prediction of Flights. *IEEE Access*, 7:181565–181579, 2019a. 3

L. Belcastro, F. Marozzo, D. Talia, and P. Trunfio. Using scalable data mining for predicting flight delays. *ACM Transactions on Intelligent Systems and Technology*, 8(1), 2016. 3

L. Moreira, C. Dantas, L. Oliveira, J. Soares, and E. Ogasawara. On Evaluating Data Preprocessing Methods for Machine Learning Models for Flight Delays. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2018-July, 2018a. 3, 4

Jiawei Han, Jian Pei, and Micheline Kamber. *Data Mining: Concepts and Techniques*. Elsevier, June 2011. ISBN 978-0-12-381480-7. 3, 4, 17

J.J. Rebollo and H. Balakrishnan. Characterization and prediction of air traffic delays. *Transportation Research Part C: Emerging Technologies*, 44:231–241, 2014. 3

J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under Concept Drift: A Review. *IEEE Transactions on Knowledge and Data Engineering*, 2018. 5

T.R. Hoens, R. Polikar, and N.V. Chawla. Learning from streaming data with concept drift and imbalance: An overview. *Progress in Artificial Intelligence*, 1(1):89–101, 2012. 5

A. Cavacini. What is the best database for computer science journal articles? *Scientometrics*, 102(3):2059–2071, 2015. doi: 10.1007/s11192-014-1506-1. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-84925539422&doi=10.1007%2fs11192-014-1506-1&partnerID=40&md5=13c2e53cadb6fa0f54763fd6ab46208c`. cited By 26. 7

H. Chen, J. Wang, and X. Yan. A fuzzy support vector machine with weighted margin for flight delay early warning. In *Proceedings - 5th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2008*, volume 3, pages 331–335, 2008. 8

S. Khanmohammadi, C.-A. Chou, III Lewis, H.W., and D. Elias. A systems approach for scheduling aircraft landings in JFK airport. In *IEEE International Conference on Fuzzy Systems*, pages 1578–1585, 2014. 8

H. Alonso and A. Loureiro. Predicting flight departure delay at porto airport: A preliminary study. In *IJCCI 2015 - Proceedings of the 7th International Joint Conference on Computational Intelligence*, volume 3, pages 93–98, 2015. 8

A. Munoz Hernandez, D. Scarlatti, and P. Costas. Real-Time Estimated Time of Arrival Prediction System using Historical Surveillance Data. In *Proceedings - 45th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2019*, pages 174–177, 2019. 8

K. Wang, J. Li, and Y. Tian. Airport Delay Prediction Method based on Improved Weather Impacted Traffic Index. In *Proceedings of 2019 IEEE 1st International Conference on Civil Aviation Safety and Information Technology, ICCASIT 2019*, pages 73–78, 2019. 8

Y. Ai, W. Pan, C. Yang, D. Wu, and J. Tang. A deep learning approach to predict the spatial and temporal distribution of flight delay in network. *Journal of Intelligent and Fuzzy Systems*, 37(5): 6029–6037, 2019. 8

Y. Yanying, H. Mo, and L. Haifeng. A classification prediction analysis of flight cancellation based on spark. volume 162, pages 480–486, 2019. doi: 10.1016/j.procs.2019.12.014. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85081056959&doi=10.1016%2fj.procs.2019.12.014&partnerID=40&md5=45b2c51485803d224053b38988951b86`. cited By 3. 10

R. Nigam and K. Govinda. Cloud based flight delay prediction using logistic regression. pages 662–667, 2018. doi: 10.1109/ISS1.2017.8389254. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85050022592&doi=10.1109%2fISS1.2017.8389254&partnerID=40&md5=5aad440442c5766a0bbdf1e345b74cb0`. cited By 2. 10

J. Chen and M. Li. Chained predictions of flight delay using machine learning. 2019. doi: 10.2514/6.2019-1661. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85083943976&doi=10.2514%2f6.2019-1661&partnerID=40&md5=9576e24099d1dc8e5fd0f5b3f09306c2`. cited By 7. 10

S. Choi, Y.J. Kim, S. Briceno, and D. Mavris. Prediction of weather-induced airline delays based on machine learning algorithms. volume 2016-December, 2016. doi: 10.1109/DASC.2016.7777956. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85009516215&doi=10.1109%2fDASC.2016.7777956&partnerID=40&md5=96307be62421a2b1f122868f353bfa43`. cited By 53. 10

S. Choi, Y.J. Kim, S. Briceno, and D. Mavris. Cost-sensitive prediction of airline delays using machine learning. volume 2017-September, 2017. doi: 10.1109/DASC.2017.8102035. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85040983840&doi=10.1109%2fDASC.2017.8102035&partnerID=40&md5=78195b7b08b523d588bec3a58f79aa47`. cited By 5. 10

N.L. Kalyani, G. Jeshmitha, B.S.U. Sai, M. Samanvitha, J. Mahesh, and B.V. Kiranmayee. Machine learning model - based prediction of flight delay. pages 577–581, 2020. doi: 10.1109/I-SMAC49090.2020.9243339. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85097838583&doi=10.1109%2fI-SMAC49090.2020.9243339&partnerID=40&md5=cb0c2b8651bba2bf3cc209c8f76550cc`. cited By 0. 10

K.R. Chandramouleeswaran, D. Krzemien, K. Burns, and H.T. Tran. Machine learning prediction of airport delays in the us air transportation network. 2018. doi: 10.2514/6.2018-3672. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85051661937&doi=10.2514%2f6.2018-3672&partnerID=40&md5=449aa8627fcee6d92eaa77c5964872c0`. cited By 3. 10

B. Thiagarajan, L. Srinivasan, A.V. Sharma, D. Sreekanthan, and V. Vijayaraghavan. A machine learning approach for prediction of on-time performance of flights. volume 2017-September, 2017. doi: 10.1109/DASC.2017.8102138. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85040961551&doi=10.1109%2fDASC.2017.8102138&partnerID=40&md5=a6f1d279b0a9801021dee43dcc21cd27`. cited By 12. 10

G. Gui, F. Liu, J. Sun, J. Yang, Z. Zhou, and D. Zhao. Flight delay prediction based on aviation big data and machine learning. *IEEE Transactions on Vehicular Technology*, 69(1):140–150, 2020b. doi: 10.1109/TVT.2019.2954094. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-85078459656&doi=10.1109%`

2fTVT.2019.2954094&partnerID=40&md5=ce9eb38d09eb08971ebb495c1abaabba. cited By 50. 10

F. Liu, J. Sun, M. Liu, J. Yang, and G. Gui. Generalized flight delay prediction method using gradient boosting decision tree. volume 2020-May, 2020. doi: 10.1109/VTC2020-Spring48590.2020.9129110. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85088318227&doi=10.1109%2fVTC2020-Spring48590.2020.9129110&partnerID=40&md5=1e08dd38152feb234fcbf5926f6fdd88. cited By 2. 10

R. Arnaldo Scarpel and L.C. Pelicioni. A data analytics approach for anticipating congested days at the são paulo international airport. *Journal of Air Transport Management*, 72:1–10, 2018. doi: 10.1016/j.jairtraman.2018.07.002. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85049917848&doi=10.1016%2fj.jairtraman.2018.07.002&partnerID=40&md5=76f4d8e1866f587f71ddc38d74ff5dc1. cited By 2. 10

L. Moreira, C. Dantas, L. Oliveira, J. Soares, and E. Ogasawara. On evaluating data preprocessing methods for machine learning models for flight delays. volume 2018-July, 2018b. doi: 10.1109/IJCNN.2018.8489294. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85056526580&doi=10.1109%2fIJCNN.2018.8489294&partnerID=40&md5=29ee55194a2b437c926912622327e326. cited By 4. 10

H. Khaksar and A. Sheikholeslami. Airline delay prediction by machine learning algorithms. *Scientia Iranica*, 26(5 A):2689–2702, 2019. doi: 10.24200/sci.2017.20020. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85073448549&doi=10.24200%2fsci.2017.20020&partnerID=40&md5=d6fdcec58592376cd264b37b5d29c8a4. cited By 4. 10

F. Rehm and F. Klawonn. Learning methods for air traffic management. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3571 LNAI:992–1001, 2005. doi: 10.1007/11518655_83. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-26944450880&doi=10.1007%2f11518655_83&partnerID=40&md5=f5b1b564d7a0faca4a7ab3f87ae53f0d. cited By 12. 10

Y. Guleria, Q. Cai, S. Alam, and L. Li. A multi-agent approach for reactionary delay prediction of flights. *IEEE Access*, 7:181565–181579, 2019b. doi: 10.1109/ACCESS.2019.2957874. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85077972797&doi=10.1109%2fACCESS.2019.2957874&partnerID=40&md5=ddf0992d1145c642d572ac915a5b3093. cited By 4. 10

B.W. Yap and C.H. Sim. Comparisons of various types of normality tests. *Journal of Statistical Computation and Simulation*, 81(12):2141–2155, 2011. 14

C. Schaffer. Technical note: Selecting a classification method by cross-validation. *Machine Learning*, 13(1):135–143, 1993. doi: 10.1023/A:1022639714137. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-0000245470&doi=10.1023%2fA%3a1022639714137&partnerID=40&md5=b5c3990590e2c8bf4eb75f02156085af`. cited By 191. 17

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer, 1 edition, August 2013. ISBN 978-1-4614-7137-0. 17

Jürgen Groß. *Linear regression*, volume 175. Springer Science & Business Media, 2012. 20