Les Cahiers du GERAD ISSN: 0711–2440

Comptes rendus du Dixième atelier de résolution de problèmes industriels de Montréal, 13–27 août 2020

Proceedings of the Tenth Montréal Industrial Problem Solving Workshop, August 13–27, 2020

Odile Marcotte, éditrice

G–2021–51

Septembre 2021

La collection *Les Cahiers du GERAD* est constituée des travaux de recherche menés par nos membres. La plupart de ces documents de travail a été soumis à des revues avec comité de révision. Lorsqu'un document est accepté et publié, le pdf original est retiré si c'est nécessaire et un lien vers l'article publié est ajouté.

Citation suggérée : Odile Marcotte, éditrice (Septembre 2021). textcolorredComptes rendus du Dixième atelier de résolution de problèmes industriels de Montréal, 13–27 août 2020 / Proceedings of the Tenth Montréal Industrial Problem Solving Workshop, August 13–27, 2020, Rapport technique, Les Cahiers du GERAD G–2021–51, GERAD, HEC Montréal, Canada.

Avant de citer ce rapport technique, veuillez visiter notre site Web (https://www.gerad.ca/fr/papers/G-2021-51) afin de mettre à jour vos données de référence, s'il a été publié dans une revue scientifique.

The series *Les Cahiers du GERAD* consists of working papers carried out by our members. Most of these pre-prints have been submitted to peer-reviewed journals. When accepted and published, if necessary, the original pdf is removed and a link to the published article is added.

Suggested citation: Odile Marcotte, éditrice (September 2021). textcolorredComptes rendus du Dixième atelier de résolution de problèmes industriels de Montréal, 13–27 août 2020 / Proceedings of the Tenth Montréal Industrial Problem Solving Workshop, August 13–27, 2020, Technical report, Les Cahiers du GERAD G–2021–51, GERAD, HEC Montréal, Canada.

Before citing this technical report, please visit our website (https://www.gerad.ca/en/papers/G-2021-51) to update your reference data, if it has been published in a scientific journal.

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2021
– Bibliothèque et Archives Canada, 2021

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2021
– Library and Archives Canada, 2021

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Préface

Le Dixième atelier de résolution de problèmes industriels de Montréal, qui eut lieu du 13 au 27 août 2020, fut organisé conjointement par le Centre de recherches mathématiques (CRM) et l'Institut de valorisation des données (IVADO). La préparation de l'atelier fut marquée par le début de la pandémie et le comité organisateur (incluant Margarida Carvalho, Fabrizio Gotti, Nancy Laramée, Odile Marcotte, Jean-Marc Rousseau, Juliana Shulz et Guy Wolf) décida que l'atelier se tiendrait en mode virtuel. Plus de 60 personnes s'inscrivirent à l'atelier et examinèrent quatre problèmes, fournis respectivement par Air Canada, Desjardins, Hydro-Québec et l'IATA. Je remercie chaleureusement ces partenaires, les coordonnateurs des équipes, Philippe Langlais, Sébastien Gambs, Yi Yang, Huaxiong Huang et Denis Larocque, ainsi que les conseillers IVADO dont la collaboration fut essentielle au bon déroulement de l'atelier. Finalement j'exprime toute ma reconnaissance à Karine Hébert, qui m'a aidée à mettre en forme ces comptes rendus.

Odile Marcotte
Professeure associée, UQAM
Membre associé, GERAD

CENTRE DE RECHERCHES MATHÉMATIQUES

IVADO

Foreword

The Tenth Montreal IPSW took place on August 13-27, 2020, and was jointly organized by the Centre de recherches mathématiques (CRM) and the Institute for Data Valorization (IVADO). The pandemic started as we were preparing the workshop and the Organizing Committee (consisting of Margarida Carvalho, Fabrizio Gotti, Nancy Laramée, Odile Marcotte, Jean-Marc Rousseau, Juliana Shulz, and Guy Wolf) decided to hold a virtual workshop. More than 60 persons registered for the workshop and studied four problems, submitted respectively by Air Canada, Desjardins, Hydro-Québec, and IATA. I thank our industrial partners, the team coordinators (Philippe Langlais, Sébastien Gambs, Yi Yang, Huaxiong Huang, and Denis Larocque), as well as the IVADO advisors whose collaboration was crucial for organizing this virtual workshop. I am also very grateful to Karine Hébert, who helped me put these proceedings together.

Odile Marcotte
Adjunct Professor, UQAM
Associate member, GERAD

CENTRE DE RECHERCHES MATHÉMATIQUES

IVADO

Contents

1 Detection of recurring defects in airline incident report

David Alfonso Hermelo [a]
Ilan Elbaz [a]
Tianjian Gao [a]
Fabrizio Gotti [a]
David Kletz
Philippe Langlais [a]
Vincent Letard [a]
Lucas Pagès [a]
Frédéric Piedboeuf [a]
Helen Samara Dos Santos [b]
Tina Yang Zhou [c]

[a] *RALI & DIRO, Université de Montréal, Montréal (Québec), Canada*

[b] *Memorial University, St. John's (Newfoundland and Labrador), Canada*

[c] *RALI & University of Bath, Bath, United Kingdom*

September 2021
Les Cahiers du GERAD

1.1 Problem description

This section is slightly adapted from the official description provided by Air Canada.

1.1.1 Context

Transport Canada mandates per the Canadian Aviation Regulation (CAR 706.05 and STD 726.05) that an Air Operator Certificate (AOC) holder must include in its maintenance control system procedures for recording and rectification of defects, including the identification of recurring defects. Defects can be classified into two distinct categories: Safety/Airworthy related defects and Non-Safety/Airworthy related defects.

1. Safety/Airworthy defects are covered under the Minimum Equipment List (MEL), a document approved by the Minister pursuant to CAR 605.07 (3) that authorizes an operator to operate an aircraft with aircraft equipment that is inoperative under the conditions specified therein; the MEL may specify that certain equipment must be operative. Each MEL has its own unique identifier and each MEL-type defect has an Air Transport Association (ATA) technical classification.
2. Non-MEL defects are defects that are raised for items that are not Safety/Airworthy related, such as scratches or gauges on surfaces, among many more designated classes. Each defect has an Air Transport Association (ATA) technical classification.

Recurring defects are the focus of the current problem.

1.1.2 The problem

The ATA defect classification is carried out manually in real time by the engineer, flight attendant, or pilot on board. The ATA classification tables have generic identifiers such as 25-00-00, which is labelled "Cabin General". This means that any defect that occurs in the cabin can technically be classified as such, making the effort of tracking recurring non-MEL defects onerous. Since there are hundreds of different combinations in the ATA classification categories and thousands of employees reporting defects, the probability of defects being reported with the required ATA classification standard (apart from the generic classification) is low. As a result, the ATA category numbers cannot be considered as a unique identifier for the purpose of tracking recurring non-MEL defects. Another problem is the wide presence of synonyms and acronyms while describing defects: for example, "Nose Landing Gear," "Nose Gear," or even "NLG" may refer to the same type of defect. Consequently the classification has had to be carried out manually on the basis of the defect descriptions, which is again time-consuming and arduous.

1.1.3 Desired solution

Air Canada Maintenance wishes to detect recurring defects automatically in a way that meets and exceeds Transport Canada requirements for both MEL and Non-MEL defects. Defects are considered recurring if a failure mode is repeated 3 times, on an aircraft, within 15 flight segments of a previous repair made with respect to that failure mode. For this workshop, the goal was slightly reframed and we strived to detect in an automatic fashion recurring intervals of 3 defects within 30 days, 4 within 40 days, and 5 within 50 days. Additionally, Air Canada desires to relabel reports with ATA Chapter/Section labels in a more exact way, in an effort to sanitize the data set. To carry out these tasks, Air Canada provided a large data set of defect reports, including MEL, textual defect description, ATA labels, aircraft tail number, etc. Auxiliary data was also provided, including reference tables of acronyms and synonyms used in the airline industry.

The next sections are organized as follows. We present in Section 1.2 the Air Canada data set that we worked on. We describe in Section 1.3 the normalization techniques implemented. In Section 1.4, we report the experiments conducted for assigning a defect its ATA code, while in Section 1.5 we relate our efforts to detect recurrent defects. Finally we discuss in Section 1.6 possible continuations of the work carried out during the workshop.

1.2 Data

Air Canada provided the team with a corpus of logbooks of aircraft defects reported by different Air Canada employees (technicians, cabin crews, pilots) from January 2018 to December 2019. We had access to those defects as a spreadsheet with various fields (48 in total) of different data types describing each defect. Prior to the workshop, Fabrizio Gotti sanitized the data and created a GitHub repository containing sample Python scripts illustrating how to load the data and perform a few simple operations. This has greatly helped the team start digging into the enormous data. Keith Dugas also provided many explanations on the data fields in the weeks leading to the workshop. These explanations led to additional (and valuable) documentation being made available to the workshop participants.

Due to the time constraint and the difficulty of understanding all the intricacies of the data fields, we focused on a small subset of the features:

defect_type describes the origin of the defect report. **L**: The aircraft defect logbook is used to record any technical defects of the aircraft as relates to the technical dispatch of the aircraft and/or any safety of flight items. These items reported from the flight deck are more serious. **C and E**: Cabin defect logbook used to record defects with respect to the status of the passenger cabin. **E** indicates electronic transcript of a paper logbook. **E** description is generated automatically. These codes derive their names from the following explanation: C: cabin defect logbook, E: electronic (transcript) cabin defect logbook, L: aircraft defect log book.

L-defects are considered more accurate than C defects and type E defects are considered very reliable. Unfortunately, E-type defects are overall rare in the corpus (0.3%), the majority of defects (60.1%) being of type C;

defect_description a short textual description of the defect;

ac aircraft code (aircraft manufacturer and series, obfuscated). This uniquely designates a particular aircraft in the fleet. This code designates the particular plane the report was created for;

reported_datetime date and time of the report;

chapter first-level classification of the defect according to ATA code;

section secondary classification according to ATA code; chapter and section define what is referred to as the ATA code hereafter;

recurrent the clustering output of a system (TRAX) deployed at Air Canada trying to detect recurrent defects;

resolution_description a short description of the defect resolution written by the maintenance technician/engineer.

An excerpt of the corpus is shown in Figure 1.1. The defect descriptions are in uppercase and contain jargon including acronyms, terms, seat numbers, etc. Evidently, the descriptions are typically short, as are the resolution descriptions.

Type	Description	Timestamp	Chapter	Section	MEL	Resolution
C	C/M POS 104 NEEDS RELAMPING FOR "HOT PLATE ON".	2018-01-04 02:13:00	25	0		RELAMPED.
E	GALLEY, WALL PANEL, LAMINATES, SCRATCHED, AT LOCATION:AFT, AT POSITION:FELL OFF THE WALL	2019-01-17 15:03:00	25	30		REPAIRED OK FOR SERVICE.
L	ON MORNING POWER UP HMU ADVISORY MSG APPEARED.	2019-12-28 13:13:00	46	0	491753	TRIAGE

Figure 1.1: Excerpt of the data set of defect reports provided by Air Canada over the period 2018-2019. Each defect is composed of 48 fields, among which a type indicating the logbook type of the defect, its description (a short text), the time when it was reported, as well as its ATA code (a chapter and a section, which together refer to a predefined node in the ATA taxonomy of defects).

The ATA label (chapter-section) provides important information characterizing a defect, even if it is only partially reliable, as stated in the official description provided by Air Canada. Therefore we report in Figure 1.2 the distribution of defect chapter labels as well as the distribution of sections within a chapter. We observe that some chapters appear much more frequently than other chapters, Chapter 25 (which indicates cabin incidents) being the most frequent. Also the section distribution within a chapter is not uniform. Among the Chapter-25 defects, Section 20 is the most frequent. The ATA code 25–20 corresponds to issues with passenger convenience items, non-essential equipment and furnishings, flight attendant seats, passenger seats, and non-essential storage equipment.[1] We also observe that an important number of defects (18.8% of the full data set) is associated with section 0, a non-informative section used as a "catch-all" section when the maintenance person fails to identify the section specific to the problem reported.

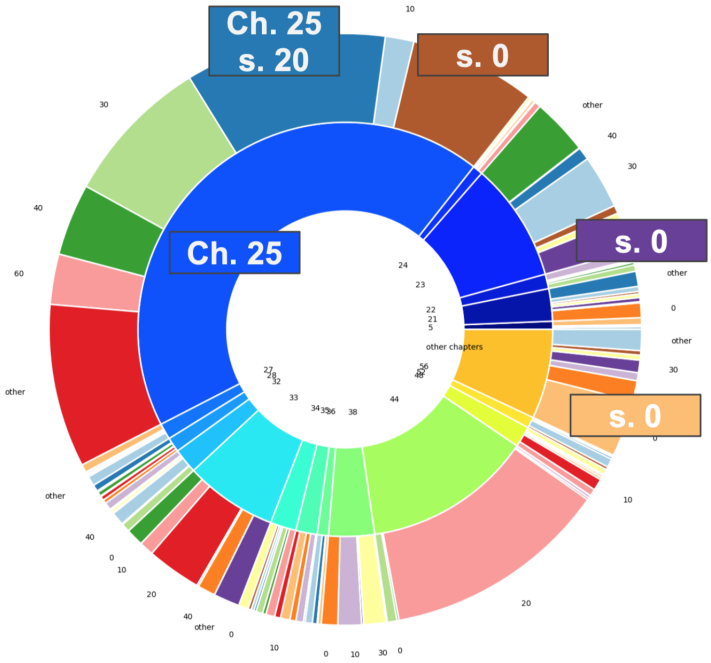

Figure 1.2: Distribution of ATA codes (chapter-section) in the corpus provided. The inner circle of the pie chart indicates the chapter distribution, while the outer circle represents the section distribution. The most frequent ATA code is 25–20, characterizing issues with passenger convenience items, non-essential equipment and furnishings, flight attendant seats, passenger seats, and non-essential storage equipment.

1.2.1 Subsets

We partitioned the original data into 3 subsets of which the main characteristics are reported in Table 1.1: FULL, containing 460k defects reported by various persons, and therefore containing a high level of noise; TRAX, gathering 47k defects that were concerned by the TRAX system in place at Air Canada;[2] and RELIABLE, which contains 34K defects that are believed to be the most reliable according to our understanding of the data set.[3] We split along the aircraft boundaries each subset into a training, validation, and test parts. The details of each data set are reported in Table 1.1.

[1] https://tc.canada.ca/en/aviation/aircraft-airworthiness/master-minimum-equipment-list-mmel/ata-25-equipment-furnishing

[2] The TRAX system identifies 14.2k clusters covering 47k of the defects in the full corpus, for an average cluster size of 3.3 defects per cluster. Initially, we thought that this was the ground truth for the clustering task, but we came to understand during the workshop that this isn't so. On the contrary these clusters are only a rough classification and are reviewed by humans after they are created by TRAX. This automated system clusters defects based on very simple rules,

Table 1.1: Main characteristics of the data sets used to benchmark solutions. The number of defects refers to the total number of defects used in the data set, the number of token types refers to the total number of different labels (chapter-section combinations), the average description length is the average number of characters in the defect description (including, within parentheses, the number of words, following a simple space splitting), and finally the number of % of section-0 defects refers to the percentage of defects that have 0 as a section, which we remove for training and testing.

	#defects	#token types	avr. desc. length	% section-0 defects
		FULL		
train	380209	736	63.4 (11.7)	18.8
valid	33465	510	63.9 (11.6)	20.2
test	46920	521	64.2 (11.8)	19.8
		TRAX		
train	28309	363	85.1 (15.4)	26.6
valid	9436	304	85.1 (15.5)	26.8
test	9437	299	84.8 (15.3)	26.5
		RELIABLE		
train	29220	116	105.7 (18.9)	0.12
valid	2897	97	109.1 (18.9)	0.38
test	2317	92	100.9 (17.7)	0.22

1.3 Data normalization

As is often the case with real data, we rapidly noticed a large number of words containing spelling mistakes, abbreviations, jargon, or acronyms. To give a sense of the kind of noise,[4] we report in the inner pie chart of Figure 1.3 the proportion of token types[5] that are listed in an in-house lexicon gathering 370 107 English words (A): only 12.5% of token types present in the defect descriptions belong to the lexicon, we call them known words. The vast majority (B) of token types are indeed unknown. The outer pie chart further refines the categorization of unknown words into acronyms (c), airport codes (d), abbreviations (e), and words containing at least one digit (f). For compiling those broad statistics, we had at our disposal a list of 5 328 airport codes (e.g. *SAP* for *SAN PEDRO SULA*), 12 288 abbreviations (e.g. *MONG* for *MONITORING*), and 2 188 acronyms (e.g. *ACFT* for *aircraft*).

Part of the team therefore spent some time investigating different normalization methods of such material, which we will report later on.

1.3.1 Acronym detection

Even if a dedicated website had been prepared before the workshop, with all the useful information listed (including a rather large list of acronyms), one member of the team[6] did investigate whether acronyms (e.g. *AVOD*) and their possible plain forms (e.g. *AUDIO/VIDEO ON DEMAND*) could be mined directly from the textual description of the defects.

We searched in all the descriptions for bracketed sequences of letters,[7] then output the n preceding words as a context into which we searched for possible resolutions. Identifying candidate resolutions for an acronym can be carried out by aligning the letters in the acronyms with those of the context. Often many alignments are possible. Therefore we assigned a score to each alignment in order to favour those

most important of all is the equality of the ATA codes between two defects. TRAX also factors in the defect timestamp in order to create clusters of various levels of recurrence (1, 2, 3 depending on the timespan encompassed by a cluster).

[3] We removed C-type defects because they are less reliable. For the other types, we replaced the chapter and section of the provided ATA code thanks to a mapping from the MEL code that was explained to us by Air Canada.

[4] We call it noise with the perspective of a model, but the data has nothing wrong in it, it is simply the way it is!

[5] We distinguish a token type (a word form) from its occurrences in a corpus. Token types are defined here according to the `word_tokenize` function from the NLTK library.

[6] Contact person of the present report that feels ashamed not having noticed the already large acronym list available ...

[7] We used a simple regular expression for this, insuring the sequence contained at least 2 and at most 5 characters, two metaparameters that were not investigated.

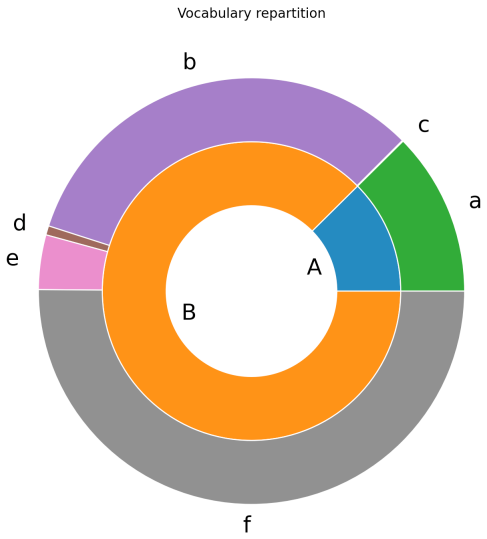

Figure 1.3: Distribution of token types in the description field of the Full data set. The inner circle identifies two categories of token types: those listed in an in-house English lexicon (A, 12.5%) and the other ones (B). The outer circle refines the distribution by distinguishing token types listed in dedicated resources as acronyms (c), airport codes (d), abbreviations (e), and words with at least one digit (f). Thus section (b) identifies token types unknown from our lists of tokens, while section (a) still represents the proportion of known token types in our English lexicon.

where the aligned letters in the context are the first letters of words. We output the m best scored resolutions for a given context, provided they received a decent enough score. Since different defect descriptions use similar acronyms (possibly with different contexts), we get a distribution of acronyms and their resolutions.

This process is depicted in Figure 1.4 for the acronym *FAP* found 37 times within parentheses in the defect descriptions of the training material of the FULL data set. On this data set, with m and n set to 5 and 3 respectively, we identified 4 146 pairs of acronym/left context pairs, involving 558 different acronyms. Once resolved, this led to 665 acronym/resolution pairs, involving 202 acronyms (some acronyms may have different resolutions, as illustrated in Figure 1.4). Table 1.2 shows the 5 most frequent candidates, as well as the 5 less frequent ones. Some candidate resolutions are clearly wrong, such as the last one. Filtering is of course possible, but we did not explore this.

input

▷ L2 DOOR LOW PRESSURE. (CHECK DOOR PRESSURE MESSAGE ON THE FLIGHT ATTENDANT PANEL **(FAP)**)
▷ (CABIN DOOR CHECK SLIDE PRESSURE MESSAGE ON FLIGHT ATTENDANT PANEL **(FAP)**)
▷ FWD. F/A PANEL **(FAP)** CIDS "CAUTION" LIGHT WENT ON.
▷ SCREEN FOR CABIN LIGHT CONTROL **(FAP)** R5 AND L2 WITH BLACK IMAGE
...

acronym / left context

FAP	ON THE **F**LIGHT **A**TTENDANT **P**ANEL
FAP	MESSAGE ON **F**LIGHT **A**TTENDANT **P**ANEL
FAP	FWD. **F**/ **A** **P**ANEL
FAP	. SCREEN FOR CABIN LIGHT CONTROL
FAP	TANK INFO ON CIDS PANEL

resolutions

2	FLIGHT ATTENDANT PANEL
1	F/A PANEL

Figure 1.4: Illustration of detection of candidate resolutions for the (potentiel) acronym FAP. 37 defect descriptions in the training part of the Full data set contain the mention (FAP) (we show 4 of them in the top box) leading to 5 different left contexts (middle box), whose resolution leads to 2 candidates (bottom box).

Since our data set also contains a column `resolution_description`, we also applied our procedure to this material from which we could extract other acronyms and their resolutions. We identified 61 new pairs (33 new acronyms). The description in the resolution column is much more standardized, and often, acronyms are used without being mentioned within parentheses.

Because we have access to a list of 2 871 acronyms/resolution pairs (2 144 acronyms) we can use this reference to evaluate our process. We can also check whether the automatic process finds acronyms that were not previously listed. Out of the 235 acronyms we found, 85 were already listed as acronyms in the reference list, 150 were not. While we are not able to judge the validity of the new acronyms discovered, a random inspection of them seems to indicate that they are mostly good acronyms. The ones marked by a star in Table 1.2 are actually new acronyms. Again, the last one is an error of our extraction procedure (that would be easy to filter). It is of course tempting to evaluate the 85 acronyms we identified that are already in our reference list, but this turns out to require human intervention because the reference list often contains some annotations that must be removed before comparing the lists. Suffice it to say that most acronyms we found are actually correctly resolved, sometimes with minor variations. We found some cases where the resolution identified automatically is fairly different from the reference one, as for *CAM* in Table 1.3.

Table 1.2: The 5 most frequent acronym/resolution pairs in the defect description field of the Full corpus, as well as the 5 less frequent ones. The letter alignment (the best scored one) is indicated in bold. Acronyms marked by a star are not listed in our reference list.

freq.	acronym	candidate resolution
915	AVOD⋆	**A**UDIO/**V**IDEO **O**N **D**EMAND
306	EFB	**E**LECTRONIC **F**LIGHT **B**AG
223	IFE	**I**N-**F**LIGHT **E**NTERTAINMENT
182	ICS⋆	**I**NTEGRATED **C**OOLING **S**YSTEM
170	APU	**A**UXILIARY **P**OWER **U**NIT
1	TA	**T**RAFFIC **A**LERT
1	TAT	**T**OTAL **A**IR **T**EMPERATURE
1	TCP ⋆	**T**UNING AND **C**ONTROL **P**ANELS
1	VFSG⋆	**V**ARIABLE **F**REQUENCY **S**TARTER **G**ENERATOR
1	WALL⋆	RO**W**45 **A**ND 46 **L**IGHT PN**L**

Table 1.3: Excerpt of acronyms and resolutions identified automatically (CAN), and their corresponding resolution in our reference list (REF).

ADF	CAN	AUTOMATIC DIRECTION FINDING
	REF	Australian Defence Force
	REF	Automatic Direction Finding (equipment)
	REF	Automatic direction finder
AIP	CAN	ATTENDANT INDICATION PANEL
	CAN	ATTENDANT INDICATION PANELS
	REF	Aeronautical Information Publication
CAM	CAN	CABIN ASSIGNMENT MODULE
	REF	Cockpit area microphone (part of the cockpit voice recorder)

1.3.2 Spell checking

Without much surprise, the descriptions of defects are fraught with many typos. In order to identify some of them, we gathered a lexicon of 307k words (plain words and their inflected forms, including conjugations) we collected from a github repository[8] to which we added the 55k most frequent words in English Wikipedia, as well as an in-domain lexicon built by listing all the alphabetical words found in the description and resolution columns of the data set. Then for every word in every defect description and every defect resolution, we computed the list of closest words from our lexicons, according to the

[8] https://github.com/dwyl/english-words/

Levenshtein distance [5]. Actually we used the so-called Damerau-Levensthein distance[9], which views the transposition of 2 adjacent symbols as one operation (eg. *glucometer* / *gulcometer*) while this transposition would amount to 2 operations with the plain Levenshtein distance. We conservatively kept the words with a distance of at most 1 edit per 5 characters, yielding a list of 15 297 typo/correction pairs involving 5 631 different correct forms, the 5 most frequently misspelled words being reported in Figure 1.5. It is rather surprising that some words got so many faulty variants. For instance the word *intermittently* has no less than 54 variants according to our procedure. While some might be due to segmentation issues (e.g. *outintermittently*, *upaircraft*), most variants we inspected seem to be just typos. This clearly militates in favour of a unified application for typing defect reports.

word	# typos	5 randomly picked typos
intermittently	54	intrmittently, inttermittently, ntermittently, outintermittently, intermittentally
illuminated	51	illuuminated, iluminated, ilumminated, immuminated, innluminated
glucometer	44	gluscmeter, glvcometer, glycometer, glyvometer, gucometer
aircraft	43	aircrqaft, aircrtaft, upaircraft, aircvraft, airfcraft
working	41	workign, workiing, workimg, worlking, worrking

Figure 1.5: Five most frequent words that got misspelled according to the automatic procedure described, the number of different typos identified, as well as 5 randomly picked ones. Keep in mind that some typos are due to a tokenization issue, and that we may wrongly associate a typo to a given form.

1.4 Classification of defects

This section is concerned with the automatic classification of a report into its ATA code (chapter and section). We tried a number of typical approaches to classification that we applied to our data sets, focusing only on the defect description column, while some other columns may improve performance. The metric we report is the standard F1 score (the harmonic mean of precision and recall).[10]

1.4.1 Bag-of-words models

A strong baseline consists in representing the input (in our case the defect description) as a bag of words (bow), and then train a classifier on top of it. We ran a number of variants of a support vector (SVM) classifier.[11] More precisely, the defect description is represented into a huge sparse vector whose dimension equals the number of different units in the descriptions of the training part. The coefficient associated with each dimension is the so-called tf-idf score, which favours frequent units while downgrading those that are present in too many descriptions.[12] We considered different types of units, among which words, ngrams of words, and ngrams of characters. Since this kind of representation can be quite large, we also considered variants where only the most frequent units are kept in the bow representation, but filtering typically comes at a price in performance.[13]

1.4.2 Deep learning models

We also tested a number of deep learning approaches. The approach consisting of fine-tuning a pre-trained BERT model [4] on the training material available is nowadays ubiquitous in NLP, since the authors reported impressive results in doing so for a number of challenging benchmarks. It is worth noting that this is a much heavier approach: fine-tuning BERT requires 30 minutes per epoch on the Reliable data set and 15 hours on Full for a computer equipped with a GTX 1070 GPU, while an SVC model is typically trained within a few minutes on a laptop CPU, if not less, depending on the variants.

[9] https://fr.wikipedia.org/wiki/Distance_de_Damerau-Levenshtein

[10] https://en.wikipedia.org/wiki/F1_score. Other metrics, such as accuracy, did not yield results that were much different.

[11] We used the SVC implementation of scikit-learn.

[12] See https://scikit-learn.org/stable/modules/feature_extraction.html#text-feature-extraction for more.

[13] For instance, on the Trax benchmark, the SVC computed on the bow of all unigrams and bigrams yields an F1 score of 81.1, while keeping the most 5k (resp. 1k) frequent ngrams yields a score of 79.2 (resp. 75.7).

Unfortunately, fine-tuning BERT was not successful in our case.[14] While we did not have time to perform in-depth analysis of the reasons why BERT failed to learn, we believe that it is mostly due to the label distribution. It is a well-known problem that backpropagation with an unbalanced data set tends to perform poorly, since backpropagation favors the majority classes and tends to ignore the uncommon labels. Since in the distribution of the data set the majority of the labels are uncommon, it is logical to expect that a regular neural network will not be able to learn to classify the data set efficiently.

We developed two variants to try dealing with label imbalance: oversampling and weighting the samples. Both however diminished the performances, since the network was now over-predicting uncommon classes. Time constraints prevented us from exploring more solutions, such as the focal loss.

We also tried a number of canonical deep learning approaches, including a recurrent neural network (with a GRU [3] cell), for classification, as well as simpler variants where a defect description representation is obtained by averaging pre-trained word embeddings, then fed into an SVC classifier. We considered pre-trained GloVe embeddings [7].[15] We also trained our own embeddings on the descriptions of the FULL data set, in the hope that they would capture specificities of the data (acronyms, typos, etc.). We used for this a Skip-gram model [6].[16] We also trained fastText word embeddings [1]. For some reasons, however, we did not test them for classification,[17] but used them as a (sanity) check that they capture useful information. Figure 1.6 lists the words most similar to some randomly picked words according to fastText: we observe that word embeddings behave as expected, that is, they capture words that share related meaning (synonyms, antonyms, etc.), as well as words that share similar spellings (typos, morphological variants).

screen	screeen ptv blackscreen black creen ptc sreen screeb
water	potable waterspigot nowater faucets faucett hotwater flowing
missing	broken mising missising retaining boken brokened brken
sink	clogged draining drain draing drains unclogged sinks glogged
open	close closed closing opening reopen unlatch latch unlatched

Figure 1.6: Most similar words (right) of some randomly picked words (left), according to a fastText word-embedding model trained on the descriptions of the data set.

Table 1.4: In-domain classification results (F1 scores) on our 3 benchmarks: Reliable (Rel.), Full and Trax. Due to time constraints, not all variants were tested over all benchmarks.

classifier	REL.	TRAX	FULL
SVC variants			
word 1-5 ngrams, no normalization	97.1		
word 1-3 ngrams, no normalization	97.5	80.6	
word 1-2 ngrams, no normalization	97.7	80.9	59.9
word 1-2 ngrams, spelling replacement	97.8	79.8	59.9
word 1-2 ngrams, acronym and spelling replacement	97.9	79.9	60.1
word 1-2 ngrams, nltk porter/snowball stemming	97.6	81.1	
char 2-5 ngrams, no normalization	97.5	81.8	
dummy: majority class	36.2	6.7	6.7
BERT fine-tuning, acronym and spelling replacement, number replaced	18.4		
GRU, acronym and spelling replacement, number replaced	11.3		
GloVe		50.2	
Skip-gram		57.3	

[14] It is to note also that the text had to be normalized before being used by BERT.

[15] Some normalization was applied (such as error detection and acronym resolution) in order to fit the model vocabulary in a better way.

[16] We used a window size of 5.

[17] This is a topic for future work.

1.4.3 Results

We report in Table 1.4 the results of some of those variants we tested in-domain. By this, we mean that the models are trained (or fine-tuned) on the training part of a given benchmark, and tested on the testing part of the same benchmark. Note that due to time and memory issues, some variants were not tested on all the data sets. Clearly more investigations are required to give a clear picture of the task. Overall the SVC variants are the best performing ones across all benchmarks. Normalizing the defect descriptions has no to little impact on performance, and considering ngrams of characters (which avoids text normalization) instead of ngrams of words typically results in similar or better performance, while being more much memory efficient.

The increase in perfomance while comparing the data sets with varied reliability shows that indeed the full corpus is very noisy. While the TRAXdata set is not manually verified (as is the RELIABLEdata set), we consider it more reliable since it contains only recurrent defects, which are themselves classified as recurrent based on their ATA label. That means that in order to appear in the TRAXdata set, an erroneous defect would have to have been erroneously labelled 3+ times, which reduced the number of erroneous labels that make it into the TRAXdata set. The RELIABLEdata set, which has been manually verified, is 100% reliable and the fact that we obtain such high results shows that the task itself is not very complicated but that the noise makes the classification very hard.

ATA code	ATA label
	▷ most correlated features
11-32	placard:missing
	▷ placard, placards, placcard, belongs, theres, damage, sticking, sure
21-20	distribution, distribution:inoperative
	▷ recirculation, fans, fan, gasper, recirc, installed, smell, present, recir
21-30	pressurization control, pressurization control:inoperative
	▷ auto, alt, outflow, cabin, tcn, pressure, indicator, rate, altitude, auto2
21-40	heating, heating:inoperative
	▷ heating, heater, heaters, heat, ovht, cargo, iii, duct, forward, vent
21-50	cooling
	▷ pack, cooling, conditioning, deflector, ball, exhaust, packs, fcvs, bypass
21-60	temperature control, temp control:too cold
	▷ temp, zone, compt, modulating, overboard, trim, control, temperature

Figure 1.7: Most useful words for identifying randomly picked ATA codes (chapter and section) according to a tf-idf bow logistic regression model trained on the defect descriptions of the Full data set.

We were rather surprised by the overall good performance of the bow approach on our benchmarks. We investigated why this was so by training a logistic regression (LR) model on the same tf-idf bow representation. The performance of this model is slightly less than that of the SVC classifier, but it is easier to investigate which feature was found important. We report the 10 most important words according to the LR model for some randomly picked AT codes. We observe important words often come with either morphological (e.g. *heating* / *heaters*) or typographical (e.g. *placard* / *placcard*) variants. This suggests that the model is capable of some data normalization, further explaining why the normalization we conducted was not very rewarding. Also, we observe (although it would deserve a real analysis) that words are topically distributed, and globally correspond to words we would expect based on the ATA labels.

1.5 Detecting recurrent defects

Figure 1.8 shows the time span of an RD, that is, the difference in days between the first reported day to the last reported one for each genuine RD. The majority of recurrent defects are emitted during the same day and very few span more than 11 days.

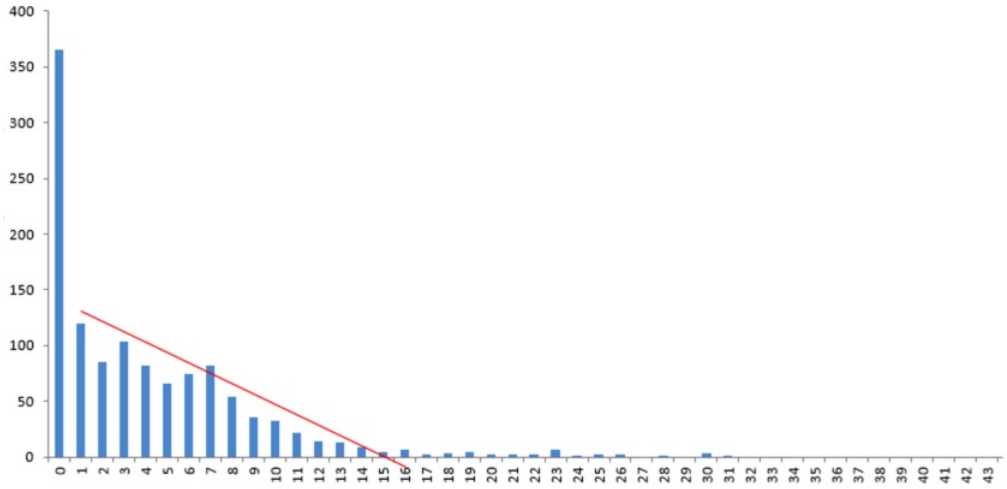

Figure 1.8: Frequency (y-axis) of the number of days (x-axis) between the first reported day to the last reported one of each manually attested recurrent event in the data set. The majority of RDs are emitted during the same day.

Figure 1.9 displays the defect descriptions associated with RD cluster 88805 (ATA code: 33–10, flight compartment). Descriptions can look rather different for a person without experience in the field.

▷ FLIGHT DECK “LT OVRD” SWITCH IS DIFFICULT TO TURN ON / OFF.
▷ DURING PDC, FOUND CAPTAIN’S DOMW LIGHT INOP.
▷ LEFT ENGINE FLOW BAR LIGHT IS U/S.
▷ LEFT ENG PRIMARY HYD. PUMP SWITCH “ON” LIGHT U/S.
▷ “L NAV ” UPPER IDENTIFICATION LT. U/S.
▷ VNAV SELECTOR SWITCH ON MCP RIGHT BUTTON OF THE SWITCH THE LIGHT BULB IS U/S.
▷ TRIM AIR SWITCH “ON” LIGHT BULB IN U/S.
▷ LT. OVERIDE SWITCH “ON” BULB U/S.

Figure 1.9: Recurrent defect 88805 encompassing 8 defects described here.

Although this is not entirely intuitive, we can view the problem of identifying RDs as clustering defects into their respective groups based on their descriptions. Defects within one group are considered recurrent. Under this view, it seems natural to evaluate the task by comparing the manual partition of defects with the one found in an automatic fashion. This is the way we are evaluating our approaches here. At the same time, most defects are not genuine recurrent defects, which suggests that the detection of RD clusters might as well be evaluated as an information retrieval task (with precision and recall measures). We leave this for future work. For each approach, we compute 4 metrics that are used for comparing two clusters:

homogeneity A decimal score in $[0, 1]$ representing the extent to which elements in the clusters found belong to the same RDs;

completeness A decimal score in $[0, 1]$ representing the extent to which elements from the same RDs are assigned to the same found clusters;

v-measure The harmonic mean of the two previous scores;

ari The adjusted Rand index is a similarity measure between both the reference and the computed clusterings. The **ari** has a value in $[-1, 1]$, 0 meaning random assignments.

1.5.1 Clustering the test material with DBScan

A straightforward approach to solving the problem at hand is clustering the defects without revisiting their original ATA classification. An appropriate algorithm for this is DBSCAN (Density-Based Spatial

Clustering of Applications with Noise),[18] which attempts to find, in an arbitrary vector space, core samples of high density then grows clusters centered on them. This method is quite interesting in our case, as it offers a natural way to find a few clusters containing only a subset of the defects in the complete data set: one only has to set a hyperparameter `eps` to limit the expansion of clusters. We experimented with different vectorial representations of defects, including a tf-idf with a latent semantic analysis (LSA) fit over the training corpus, as well as a dimension reserved for the difference in days between reported dates. We report the results below, on the test set, after exploration of the eps value on the dev set. A handy way of measuring the expansion of clusters is to measure the number of predicted clusters and their average size.

Table 1.5: Results of the recurring defects detection.

System	ARI	Homog.	Compl.	V-meas.
db-desc-tfidf-eps0.5 100-dimension tfidf with LSA	0.003	0.22	0.02	0.04
db-desc-tfidf-days-eps1.0 Same as above, + δdays	0.045	0.06	0.06	0.06
db-desc-tfidf-days-ch-eps1.0 Same as above, + δATA chapter	0.042	0.05	0.06	0.06
KMeans unigrams, 800 clusters description+resolution	0.076	0.14	0.07	0.09
DBSCAN tfidf eps 0.5 min samples 3; resolution only	0.074	0.29	0.06	0.11
SVC classifier 1-3 word ngrams + time constraint	0.10	0.04	0.05	0.028

These results are disappointing, since an ARI of 0 basically means no better than chance (1 means a perfect prediction). Nevertheless the score also has to do with the low reference quality. The text representation (tf-idf) does surprisingly little, which either suggests that it is an invalid representation, or that common textual defect descriptions are not a good indicator of their recurrence. Days elapsed between defects are much more important. A natural way to explore this algorithm further would be to add additional dimensions corresponding to additional features derived from defect metadata.

1.5.2 K-means clustering of the full data set

While the previous approach was only making use of the test material at test time, the present approach has been thought of as a means to exploit regularities in the full data set, and therefore needs all the existing data (training and test) to operate. This is definitely less handy than the previous approach, since the full data set has to be clustered. In a nutshell, we encode each defect description (or the resolution column or both) into a bow representation. We then apply the K-Means clustering algorithm[19] on those representations. Given a partition of the entire data set, we group together defects that got clustered into the same set, provided they pertain to the same aircraft and obey the time constraint given in the definition of the problem. We carried out some tuning on the training part of the Full data set, letting the number of clusters vary from 120 to 480, considering the defect_descriptions column, the resolution column, or both. This tuning was carried out in order to optimize completeness instead of the V-measure or the ARI because the Trax data, which is our reference for recurrent defects clustering, is very accurate (human review) but likely incomplete. The best performance we obtained was by considering 430 clusters, using unigrams for computing the tf-idf bag-of-words representation. It is the results of this variant that are reported in Table 1.5, and (although not very strong) this variant has the best performance overall.

[18] We used this implementation: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html.

[19] We used the implementation described at https://scikit-learn.org/stable/modules/clustering.html#homogeneity-completeness-and-v-measure.

1.5.3 Detecting by classifying defects

This approach is straightforward and is intended to serve as a baseline. We apply a classifier trained to label a defect description into its ATA code (see Section 1.4) to each defect of the test set. All defects that receive the same class label are elected recurrent defects, regardless of time constraints. Except for the ARI metric, results are very low, mainly because this approach is producing very large clusters of descriptions, which are not considered recurring defects in the reference.

1.5.4 Universal Sentence Encoder

Google [2] has released the Universal Sentence Encoder, a method intended to compute semantic similarity between any given pairs of sentences. For example, the sentence "apple is a healthy fruit" is semantically similar to "John loves eating bananas" and is dissimilar to "Honda Accord is the best family sedan". Technically speaking a language model encodes sentences and converts them into semantically-meaningful dense real-valued vectors. We tried to use such a model to group defect descriptions automatically based on semantic features in sentences such as "audio jack" and "bird strike."

We developed a simple demo of Universal Sentence Encoder and found promising results. The language model is sensitive to semantic differences, can connect similar concepts such as "audio jack" and "headphone jack," and is immune to simple typos (e.g. "screen" and "screeb"). Unfortunately we were not able to quantify the performance (clustering accuracy) of our model, which we leave for future investigations.

	recurrent_created_date	acft	defect_description	rec_id	predicted_cluster
194	2019-05-16 08:36:00	AA-2012	fuel "locator (see lm, lo)" outer transfer ope...	99566	0
195	2019-05-16 08:36:00	AA-2012	after engine start on 2 legs, right (runway sy...	99566	0
196	2019-06-11 04:01:00	AA-2012	cockpit door fault caution illuminated with cl...	100034	1
197	2019-06-11 04:01:00	AA-2012	cockpit door cloud top striker fault	100034	1
198	2019-06-11 04:01:00	AA-2012	cockpit door cloud top striker fault.	100034	1
199	2019-06-23 10:23:00	AA-2012	invoked reference l5707675 engine #1 thrust re...	100285	2
200	2019-06-23 10:23:00	AA-2012	during inspection of engine #1 thrust reverser...	100285	2

Figure 1.10: Sample clustering results using Universal Sentence Encoder.

1.6 Conclusions and future work

Our journey with Air Canada was very pleasant, generating a lot of enthusiasm from the participants as well as a few disappointments. Among them we must recognize our inability to conduct conclusive experiments on the main problem, which was to identify recurring defects within a given time frame. We believe that the task, although clear at a conceptual level, requires a much better understanding of the maintenance workflow, from the moment the defect is noticed to the moment the last recurrent instance of the defect is considered closed.

We were nevertheless luckier with our classification results, reporting very good figures with simple approaches on a subset of the clean data gathered. Again, we feel that the data contains too many varied sources of noise, and that refinements of the task (or the reference) must encompass a better understanding of the data. With that being said, we do not feel there was anything particularly unmanageable within our task: most NLP tasks of interest encompass intricacies that challenge the way the data-set is built, or the way we evaluate solutions.

Some further investigations are required to investigate a few variants we devised. In particular, we found good clustering ability of the Universal Sentence Encoder that could eventually lead to good

recurring defects detection. We also have to understand why some deep learning approaches performed so badly on the classification tasks we considered.

Given the very significant lexical corruptions of the defect descriptions and other textual elements, it could be very beneficial to Air Canada to look into leveraging spell-checking technologies, such as those already present on most computer platforms (Android, iOS, etc.). This may very well prove very cheap and would offer an invaluable return when the time comes to perform data mining and NLP manipulations on the data at hand.

At the very heart of the problem submitted by Air Canada lies the issue of label reliability. Indeed, had the ATA codes been reliably and properly attributed to defects, clustering would have been trivial, the only difficulty being one of properly taking the time frame into account when creating clusters. There are a number of remarks that can be made regarding reliability in this context. Firstly, it does seem that the ATA ontology is difficult to apply consistently. This could surely be mitigated by better formation (for instance, instructing personnel to avoid catch-all clauses), but also by using automated tools. For instance, a labeling tool not unlike those presented in this report could present the user with a list of probable labels from which he/she could pick the best code. Secondly, it may very well be that the ontology is improperly designed in the first place, leaving the maintenance personnel at a loss when labeling defects. This should be looked into, particularly for ATA combinations that are seldom used. Thirdly, the labeling task presented in this report produces an interesting by-product, in the form of a confusion matrix, i.e. a report of labels for which the human opinion and the machine output differ. This could be an interesting starting point for an investigation into improper labeling on the part of humans: the machine could very well be wrong when producing an ATA label, but if it is not, then there is a systematic problem with human labeling. Lastly, there should be a way to identify a subset of "elite maintenance personnel" whose labeling could form the base for an ultra-reliable subset of the original data. This way, an algorithm trained on these labels would benefit from the supervision of the most seasoned experts Air Canada has within its ranks, and therefore learn from the best.

Disclaimer: This report presents the work conducted during the Tenth Montreal Problem Solving Workshop for the problem submitted by Air Canada, with the much appreciated assistance of Keith Dugas and Nicholas Popovic from Air Canada. The data provided by Air Canada is extremely complex, therefore, this report, which expresses the view of the persons involved in developing solutions, might be inconclusive in a number of ways.

Bibliography

[1] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5:135–146, 2017.

[2] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. Universal sentence encoder for English. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 169–174, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[3] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In NIPS 2014 Workshop on Deep Learning, December 2014, 2014.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[5] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. Sov. Phys. Dokl., 6:707–710, 1966.

[6] Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

[7] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

2 Data anonymisation and synthesis

Mahdieh Abbasi [a]
Anne-Sophie Charest [a]
Sébastien Gambs [b]
Dena Kazerani [c]
Dylan Loader [d]
Ehsan Rezaei [e]

[a] *Département de mathématiques et de statistique, Université Laval, Sherbrooke (Québec), Canada*

[b] *Département d'informatique, Université du Québec à Montréal, Montréal (Québec), Canada*

[c] *INRIA, Paris, France*

[d] *University of Calgary, Calgary (Alberta), Canada*

[e] *Polytechnique Montréal, Montréal (Québec), Canada*

September 2021
Les Cahiers du GERAD

2.1 Introduction

Many financial institutions aim to enhance the quality of their services by extracting some data-driven insights from their customers' records using machine learning and data mining techniques. To do this they need to make such large-scale data sets available to research institutions and partners. Such data sets, however, cannot be shared directly without threatening the privacy of the individuals concerned. Therefore it is mandatory for them to anonymize their customers' information, which is often sensitive and personal, in order to protect the customers' data privacy. On the one hand the anonymized data should make it difficult or even impossible to identify the customers (participants) concerned; on the other it should preserve the statistical properties and the existing patterns of the original data. In particular it should be possible for the machine learning techniques to extract and learn these patterns for a variety of tasks (e.g. , predictive models), which leads us to the *privacy v.s. utility* dilemma.

To address this dilemma, our aim in this project is to generate *synthetic data* that retains the statistical properties of the original data (measured by a utility metric) while preserving the privacy of the participants (quantified by a privacy metric). One can generate either fully or partially synthetic data, giving rise to two types of methods. In the former, all attributes of a given data set are considered to be sensitive data: thus analysts should generate fully synthetic data to be used instead of the original data [5, 13]. In the latter, only some features are deemed to be sensitive, in which case the analysts tend either to synthesize values for these attributes [8] or to censor them [22, 25].

In this report, we discuss some widely-known or recent approaches for data synthesization, either partial or full synthesization. We also review a range of utility and privacy metrics. Then we report on the data set analysis that we carried out. Unfortunately it was not possible for us to implement fully and test a data synthesis method within the short duration of the workshop.

2.2 Methods for generating synthetic data

There are two main families of methods for carrying out synthetic data generation: *fully synthetic generation methods* and *partially synthetic generation methods*. While the former treat all the features as sensitive data that need to be synthesized, the latter aim to synthesize only the sensitive features (i.e. , those with a high risk of identity disclosure) while adding noise to non-sensitive features such that no one can infer the values of sensitive ones.

2.2.1 Fully synthetic data generation

A generative process learns the data distribution, which reflects the properties of (or the existing patterns in) the real data and is then used for generating fully synthetic new samples by drawing randomly from the data distribution learned. The objective for privacy is that the synthesized new samples cannot be mapped back to the real data, either partially or completely, and also that the original training data cannot be inferred from the generated synthetic samples.

Statistical approaches

Data imputation. In this approach, the data generation is treated as a missing data problem. More specifically, the sensitive features (or all the features) in a given data set are considered as missing data, before being imputed according to the multiple-imputation approach [32]; finally drawn random samples from these imputed populations are used to generate a synthetic data set [11].

Sampling from independent marginals. A simple baseline for generating synthetic data is sampling from the *empirical marginal distribution of each variable*, which can be estimated from the observed data. While computing these empirical distributions can be carried out using parallel computation, this method cannot capture and represent the dependencies between the variables.

Using Bayesian networks. This idea was explored a while ago in [38] and there is also PrivBayes ([39], a more recent work that satisfies differential privacy.

Deep generative models

GAN (Generative Adversarial Network) [14] and Variational auto-encoder [21] are widely-known generative deep learning models designed to estimate the data generation distribution (in the latent space) with high fidelity[1] but without any concerns for privacy. The practitioners in data privacy tend to adopt these models to generate synthetic data. It is not clear, however, whether such generative models cannot "cheat" by memorizing the training data [28], leading to a threat for privacy. Therefore, when using these models, one must ensure that the risk of identifying the training records (samples) is provably low.

GANs were originally proposed for image data sets, in which the generated samples contain real-value features. To adopt it for generating categorical, discrete, binary, and mixed data (i.e. , categorical and real-value data) features as is frequently the case in financial, health, and insurance records, Choi et al. and Camino et al. [9, 10] introduced some modifications to the original GAN. This modified GAN, called MedGAN, has been recently used for health-care data records for the purpose of generating anonymized synthetic data [13].

As the generative model of MedGAN can potentially be used and adapted to our project, we now review MedGAN in more detail. In the original GAN, the training signals from the discriminator are continuous: thus the generator can only generate continuous values instead of discrete ones. In order to generate synthetic discrete values, an encoder-decoder is integrated. The pre-trained encoder ($Enc(\cdot)$) maps each real record represented by $\mathbf{x} \in \mathcal{Z}_+^D$ (from a D-dimensional discrete-value space) into a continuous feature space, before the decoder ($Dec(\cdot)$) maps it back to the discrete-value space. The generator $G(\cdot)$ takes in a random prior $\mathbf{z}$ to generate continuous-value feature ($G(\mathbf{z})$), which is then mapped back to the discrete-value space by $Dec(G(\mathbf{z}))$. Finally, the discriminator is trained to distinguish the generated samples from the real ones.

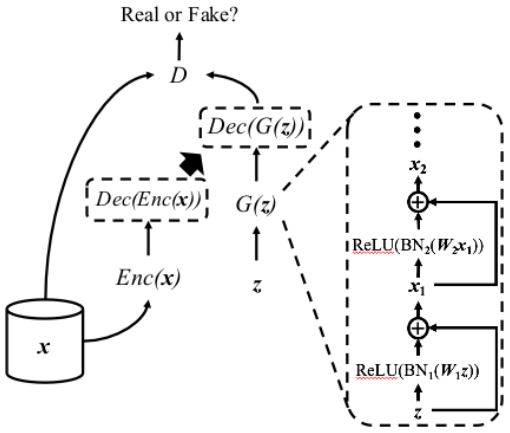

Figure 2.1: medGAN is proposed to generate discrete-value features, particularly for EHR (Electronic Health Record).

The privacy of data generated by MedGAN has been empirically assessed by different privacy metrics [10, 13], but it is better if the generative model can be explicitly trained for ensuring privacy.

[1] The synthesization is performed by randomly drawing samples from the achieved data distribution such that they are visually and statistically similar to the real (training) samples.

Abadi et al. [1] introduced several techniques, including the DP (Differential Private) Stochastic Gradient Descent algorithm, in order to achieve a privacy-preserving training of models. Using the DP-SGD, many researchers have adapted GAN or its variants [35] to ensure privacy during the training of their generative models. For instance, inspired by the MedGAN architecture and the DP-SGD, Tantipongpipat et al.proposed DP-GAN [35].

Agent-based modelling

Agent-based modelling (ABM) has been used in the context of synthesizing payments data, for instance in modelling a bank's payment processing system [12] and investigating the macroscopic impact of a disruptive event on the flow of interbank payments [3]. Synthetic data for a retail shoe store has been created using ABM by Lopez-Rojas and Axelsson [23]. This kind of generated data intrinsically respects privacy constraints if calibration is carried out manually [4]. More precisely, in [24], the authors use a simulator called RetSim and the synthethic data is the result of several iterations of the ABM algorithm based on ODD (Overview, Design Concepts, Details, [16]). Firstly entities (a different type of data) are defined and then concepts based on relations between entities are determined. The ABM model is initialized by a data format. The probability distribution of the original data is used for defining the behaviour of agents. Model parameters are then calibrated to achieve a satisfactory utility and privacy.

2.2.2 Partially synthetic data

Unlike the fully synthetic approach that tends to generate a new data set with the same distribution as that of the original data, the partially synthetic approach tends to modify the given data in order to anonymize it. This approach usually consists of two steps:

- Identify the sensitive features and remove them from the features set;
- Modify the values of the non-sensitive features so that they satisfy the utility and privacy constraints.

We will explain each step in detail.

Selecting sensitive features

When it comes to privacy, there are three terms that usually cause confusion: data masking, data-identification, and anonymization. Data masking includes techniques that remove and/or modify data with fake information. This approach is usually applied to features that identify individuals directly such as name, phone number, or email address. There are other features, however, that can be used to identify persons indirectly, such as gender, date of birth; they are called quasi-identifiers. De-identification or anonymization includes masking techniques and methods that handle the indirect data (quasi-identifiers). The goal of this approach is to keep the utility of the data while minimizing the probability of identifying individuals.

Let us take a look at the Home Credit Application data set. To anonymize this data set, removing features such as names, email addresses, phone numbers is necessary as finding a person based on these basic informations is easy. This is not enough, however. Indeed, if an adversary knows the targeted person's gender, marital status, income type, and the organization that the target person works for, he can group data by these features and get a group size of ≈ 25 on average ($max = 5252, min = 1, std = 136.5$), which is a small group compared to the original data set (containing more than $300k$ records).

Many methods have been proposed to detect quasi-identifiers by measuring the information leakage in the data set [2, 17, 27, 34, 37].

Partially sensitive data generation

As mentioned above, besides masking identifiers, de-identification methods tend to modify quasi-identifiers so that an adversary cannot identify individuals; they also retain the utility of the original data [7].

One of the popular methods for hiding identification is to mask data with adversarial noise. Adversarial noise was proposed for the first time to fool deep neural networks for image classification [15]. A nice property of adversarial noise is that a human cannot distinguish an image perturbed through adversarial noise from the original image. In [20], the adversarial noise was proposed to mask non-sensitive data such that an adversary cannot learn a classifier for predicting the values of sensitive features and users are able to learn a reasonable data model in order to predict the target label. In other words, as shown in Figure 2.2, the utility here is measured in terms of the accuracy of the model learned for predicting a target label and the privacy metric is the accuracy of an adversary's models for inferring sensitive data values.

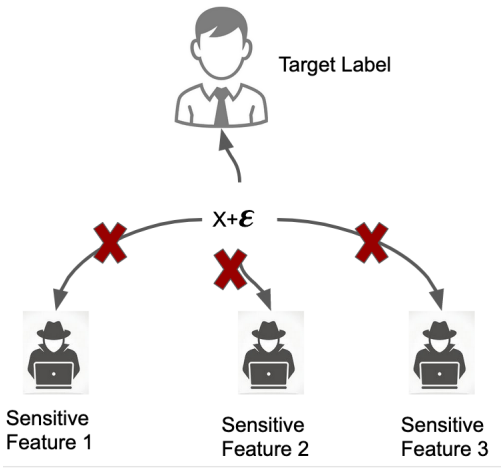

Figure 2.2: Here X denotes non-sensitive data and ϵ the adversarial noise. The adversaries try to predict the values of sensitive features while honest users only try to predict the target labels.

Learning adversarial noise

A neural network is built by creating several fully connected levels, with an activation function such as relue, sigmod, etc. Finally there is a softmax layer translating the last layer's output to the probability vector representing the probability of belonging to each class/label. A neural network is usually represented by a function $F(x, \theta)$, in which x denotes input data and θ the neural network parameters. The output of this function is the probability vector $Y = [Y_1, \cdots, Y_c]$, in which Y_i denotes the probability of input x belonging to the ith class. To learn a neural network, a loss function such as cross-entropy is optimized:

$$\min_{\theta} J(F(x, \theta), Y^*), \tag{2.1}$$

in which $J(.)$ and Y^* denote respectively a loss function and the true label of the input x. Adversarial learning methods tend to learn a small noise ϵ so that the target model is not able to classify the noisy input correctly.

$$\begin{aligned} &\min_{\epsilon} - J(F(x+\epsilon, \theta), Y^*) + \lambda \|\epsilon\|_2 \\ &argmax(F(x+\epsilon, \theta)) \neq Y^* \end{aligned} \tag{2.2}$$

To learn an adversarial noise preserving privacy, one needs to repeat the following process several times.

- Update the adversaries model (F_{Adv_i}) and user model (F_{user}) to be trained on noisy data.

$$\begin{aligned} &\min_{\theta_i} J(F_{Adv_i}(x, \theta_i), S_i^*) \quad \forall i \\ &\min_{\theta} J(F_{user}(x, \theta), Y^*) \end{aligned} \tag{2.3}$$

  Here S_i^* denotes the true value of the ith sensitive feature that the ith adversary tends to estimate.
- Learn an adversary that improves the honest user's accuracy but degrades the accuracy of the adversaries' model.

$$\min_{\epsilon} \sum_i -J(F_{adv_i}(x + \epsilon, \theta), S_i^*) + J(F_{user}(x + \epsilon, \theta), Y^*) + \lambda \|\epsilon\|_2 \tag{2.4}$$

A suitable adversarial noise, however, is a noise that can fool any other adversarial model[2], which is called transferability. To learn a transferable noise, two approaches are suggested: (i) tuning the hyperparameter of traditional methods for generating a transferable perturbation or (ii) generalizing a noise over several neural networks.

2.3 Privacy measures

There are a wide range of metrics to measure privacy of generated data, each of them concerning a different aspect of privacy.

2.3.1 Identity disclosure risk

In the fully synthetic case, the intruders aim to know whether a specific private real data record, known by the attacker, was used for training the generative model. Thus it is assumed that the intruders have access to all real data records.

Distance-based methods. To compute the membership risk for a given real data record $\mathbf{x}$, we compute whether $\mathbf{x}$ is close (as determined by a distance metric such as the Euclidean distance) to one of the generated samples. If a given real record is close enough (as indicated by a selected value for the distance) to at least one of the generated samples, we consider it as an identity disclosure risk, while otherwise it is not the case [10].

Bayesian methods. Given the intruder's prior distribution, the synthetic data, and information about the synthesis model, it is possible to attempt to compute the posterior distribution for that specific record [31]. This Bayesian framework is proposed for the categorical data sets as computing the posterior probabilities is straightforward in this case. For further reading on the Bayesian methods for measuring the disclosure risk, we refer the reader to [19]. Note that measuring risk using the Bayesian methods will depend on the assumed prior distribution by the intruder and this kind of methods is geared towards releasing multiple synthetic data sets.

2.3.2 Attribute disclosure risk

This type of risk aims to measure the chance of disclosing a sensitive attribute (or several sensitive attributes) based on a subset of attributes that are known to the attacker.

k nearest neighbors. The attacker can disclose the sensitive attributes of a given partially-known record x' by finding its k nearest neighbors from the generated synthetic data samples based on the known attributes. Then the sensitive attributes of x' are inferred by merging the corresponding sensitive

[2] As long as we can learn a good model for a honest user to predict the target label, we do not need to worry about the transferability to other good models.

attributes of these k neighbors. If the generative model memorizes the training data, the chance of this disclosure risk becomes higher. Therefore, to keep this risk low, it is crucial to discourage memorization by the generative network [10].

Prediction accuracy. With the use of a classifier trained on the synthetic data, the intruders attempt to predict some sensitive attributes of the individuals in the original data set [29].

A similar approach is used in [36], but in this case the intruder is assumed simply to use the observed conditional frequencies to predict the target, instead of creating a classifier.

2.3.3 Data-copying

Data memorization is one of the open challenges associated with generative models, such as GANs [28] and auto-encoders [30]. Also this challenge is even more crucial for privacy. Indeed data memorization can increase the identity and attribute disclosure risks. Data-copying and over-representation are two approaches that can be used to measure and detect data memorization (i.e. , overfitting) in a generative model.

Meehn et al. [26] have recently proposed a metric to measure data-copying (a form of data memorization or generating of the synthetic samples with small variations from the training ones). While the paper originally focuses on overfitting and not privacy, a synthetic data set which resembles the original data set too much is a privacy issue.

Intuitively data-copying measures distances between generated synthetic samples and both training set and original distributions. The comparison of these distances can be used to determine whether the synthetic data set can be an appropriate representation of the original distribution or the generative model is an underfitting/data-copying model.

Let χ and T denote respectively an instance space with unknown distribution P in which data points lie, and drawn samples from P, which are used to train the generative model Q. Here is the definition of data copying.

Definition 1 (Data copying) *Q is data-copying T, if in a region $\mathcal{C} \subset \chi$ and for metric $d(x) = \min_{t \in T} \|x - t\|_2^2$, generated samples from Q are closer to T than samples drawn from the original distribution P.*

$$\Delta_T(P|_{\mathcal{C}}, Q|_{\mathcal{C}}) = \Pr(B > A | B \sim L(Q|_{\mathcal{C}}), A \sim L(P|_{\mathcal{C}})) \ll \frac{1}{2}$$

in which $L(D)$ denotes the one-dimensional distribution of $d(X), X \sim D$.

Definition 2 (Over-representation) *We say that Q is over-representing P in region $\mathcal{C}$ if the probability of drawing samples from Q in this region is larger than the probability of drawing samples from P.*

$$Q(\mathcal{C}) - P(\mathcal{C}) \gg 0 \tag{2.5}$$

In over-representation, the synthetic data set is compared directly with the original distribution using the test data set. In data-copying, however, their distance from a fixed origin (test data set with samples from the original distribution) are compared with one another. Intuitively, the over-representation metric determines the performance of a generative model Q in preserving statistical features of P, while data-copying can be used for privacy goals. Note that it is possible to over-represent the data without having data-copying and vice versa. In Figure 2.3, the difference between these two concepts is illustrated.

Figure 2.4 depicts different cases for an inappropriate generated data set; region (A) and region (B) are respectively over- and under-represented (as evaluated by FID score [18] or Precision and Recall [33]), region (C) is data-copying and region (D) is underfitting. As can be seen in that figure, data-copying

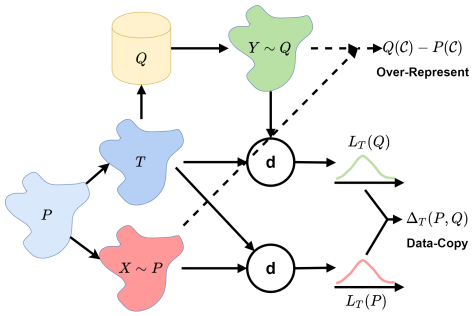

Figure 2.3: Structure of and difference between over-representation and data-copying measurement metrics.

methods should be used locally and not globally. For example, applying data-copying on a data set composed of both regions (C) and (D) would give the impression of achieving good results, while region (C) is actually data-copying. For this reason, the data space should be divided into sub-regions by using a clustering method such as k-means or DBSCAN before computing a data-copying or over-representation metric.

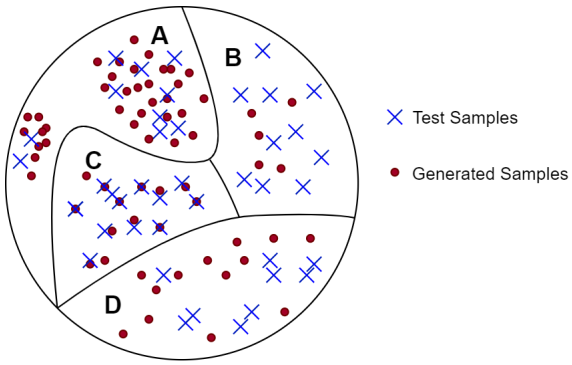

Figure 2.4: Potential situations for data synthesization.

2.4 Utility measures

The following metrics quantify to what extent the statistical properties of the real data set $\mathcal{R} = \{\mathbf{x}_i\}_{i=1}^N$ are transferred to the synthetic one $\mathcal{S} = \{\mathbf{x}'_i\}_{i=1}^N$ with N samples each. We assume that the sample complexity and data dimensionality of the real data set and the synthetic one are identical.

2.4.1 Kullback-Leibler (KL) divergence

Considering categorical features, this metric measures the discrepancy between the marginal distributions of the real and synthetic data sets with respect to a given variable (feature). This metric is able to capture the variable-wise similarity (discrepancy) but not the dependencies between variables. For a given categorical variable x, its two marginal data distributions P_x^R (for the real data set) and P_x^S (for the synthetic data set) are used to compute their KL-divergence as follows:

$$KL(P_x^R||P_x^S) = \sum_{k=1}^{|x|} P_x^R(k) \log\left(\frac{P_x^R(k)}{P_x^S(k)}\right),$$

where $|x|$ is the number of categories for the categorical variable x. The lower the KL divergence, the lower the discrepancy (i.e., the higher the similarity) between the data distributions (real and synthetic).

2.4.2 Log-cluster

Using k-means, the real data set and the synthetic one are clustered. Then the discrepancy between these two clusterings is measured in the following manner:

$$U(\mathcal{R},\mathcal{S}) = \log\left(\frac{1}{K}\sum_{k=1}^{K}\left(\frac{n_k^{\mathcal{R}}}{n_k} - \frac{n^{\mathcal{R}}}{n^{\mathcal{R}}+n^{\mathcal{S}}}\right)\right),$$

where n_k and $n_k^{\mathcal{R}}$ denote respectively the total number of samples (either from the real or synthetic data set) clustered into the kth cluster and the number of real data samples clustered into the kth cluster, while $n^{\mathcal{R}}$ and $n^{\mathcal{S}}$ denote respectively the number of real data samples and the number of synthetic data samples. Here a small log-cluster corresponds to a small difference between these two data sets.

2.4.3 Cross-classification

In cross-classification, one can train a classifier (predictive model) on a real data set $\mathcal{S}$ before testing the classifier on a held-out real test set and the synthetic samples. The ratio of performance of the model on the held-out real test set and that on the synthetic set can be considered a utility measure: the larger the ratio, the higher the similarity between $\mathcal{R}$ and $\mathcal{S}$.

2.5 Competitions and tools

NIST competition. Some public competitions were held for evaluating data synthesis methods for privacy, including one organized by NIST (the National Institute of Standards and Technology in the USA) that focused on differentially private data synthesization methods. The link to the competition can be found here: https://www.challenge.gov/challenge/differential-privacy-synthetic-data-challenge/. The results are analyzed in [6]. For the competition, contestants were provided with some training data on which to develop their methods, which were then tested on data with an identical structure. Two different data sets were used, containing both categorical and continuous variables (between 30 and 100 variables, and more than 200,000 observations). Several measures of utility covering marginal and joint distribution, as well as classification and regression tasks, were assessed. Six approaches for data synthesization were tested including PrivBayes for different values of ϵ (0.3, 1.0, and 8.0), with little impact on the utility scores.

Here are some of their conclusions.

- Non-parametric and parametric algorithms offer an implementation trade-off between requiring extensive pre-processing when using public data and requiring significant computational capabilities.
- Experimental methods, namely GANs, achieved a much lower utility than the simpler methods.

Synthpop package in R. Information about the package is available here : https://cran.r-project.org/web/packages/synthpop/index.html. This package is inspired by the methodology of imputing missing data in statistical data sets. The joint model for all variables is created by modelling each variable in turn using all other variables in the data set. A lot of different models are available, including linear and logistic regression as well as regression and classification trees. This package can handle missing values (which are simply treated as a new category, meaning that the synthetic data set may itself contain missing values), and constraints on variables (such the constraint that an individual has to be old enough to be married). This package also offers routines to produce correct statistical inference from the synthetic data.

2.6 Practical evaluation

2.6.1 Data set

The application data set used in the project includes a training set with 307,511 rows and a test set composed of 48,744 rows. Each row is characterized by 122 features: the type of 65 (resp. 41, 16) features is real (resp. integer, categorical)[3].

Table 2.1 summarizes the different categorical features.

Table 2.1: Categorical Features in the home credit risk data set.

Feature name	set size
NAME-CONTRACT-TYPE	2
CODE-GENDER	3
FLAG-OWN-CAR	2
FLAG-OWN-REALTY	2
NAME-TYPE-SUITE	7
NAME-INCOME-TYPE	8
NAME-EDUCATION-TYPE	5
NAME-FAMILY-STATUS	6
NAME-HOUSING-TYPE	6
OCCUPATION-TYPE	18
WEEKDAY-APPR-PROCESS-START	7
ORGANIZATION-TYPE	58
FONDKAPREMONT-MODE	4
HOUSETYPE-MODE	3
WALLSMATERIAL-MODE	7
EMERGENCYSTATE-MODE	2

Anomaly detection. To detect the outliers, we first want to answer this question: who are those special people who got employed 1000 years before issuance of the loan?

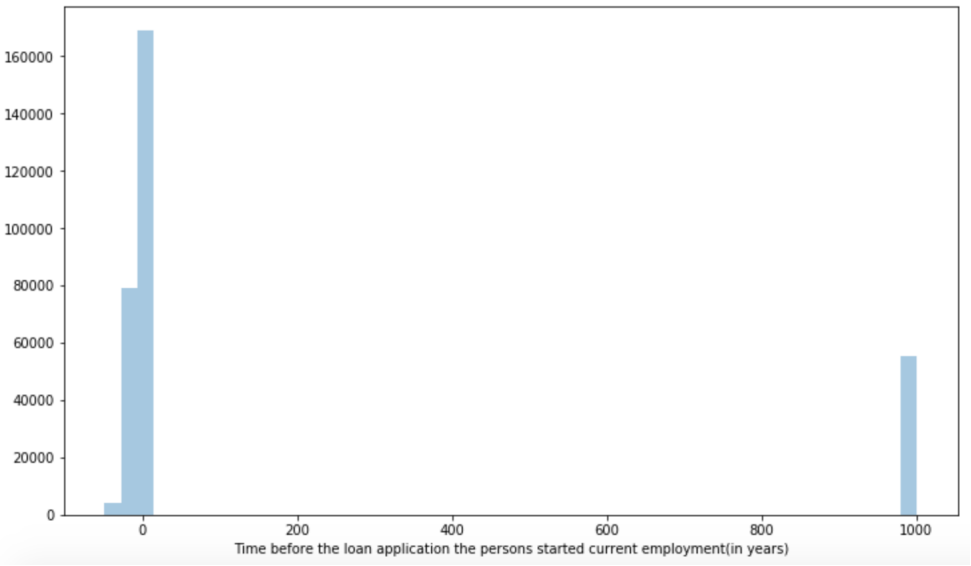

Figure 2.5: Distribution of outliers.

[3] https://github.com/rakshithvasudev/Home-Credit-Default-Risk/blob/master/Model%20Building/Home%20Credit%20Model.ipynb

In particular there are more than 50k records with applicants that got their current employment more than 100 years before the issuance of the loan. Removing all those records is not acceptable: hence we replaced the feature value for those applicants by the median one.

Accuracy as a utility metric. This data set has a target value indicating who had a difficulty in paying off his loan. More than 92% of the applicants could pay off their loan and only fewer than 8% of them could not pay.

Table 2.2: Gaussian Naive Bayes and Logistic Regression Model accuracy for each class. The average accuracy of Gaussian Naive Bayes is greater than the accuracy of Logistic Regression.

	Class	Precision	Recall	F1-Score	Support
GaussianNB	0	0.97	0.28	0.43	93362
	1	0.10	0.90	0.18	8117
LogisticRegression	0	0.92	1.00	0.96	93362
	1	0.54	0.01	0.02	8117

For this data set Gaussian Naive Bayes is more accurate than Logistic Regression while it cannot predict well who is not able to pay his loan. Stated differently, the accuracy metric is not a good metric for this data set.

Table 2.3: Gaussian Naive Bayes accuracy after removing quasi-identifiers/sensitive features.

Class	Precision	Recall	F1-Score	Support
0	0.92	0.99	0.95	282686
1	0.07	0.01	0.01	24825

Moreover, our experiments show that the accuracy of the Gaussian Naive Bayes model does not drop significantly after removing sensitive features (CODE-GENDER, NAME-INCOME-TYPE, NAME-OCCUPATION-TYPE, and NAME-FAMILY-STATUS) and also, based on non-sensitive features, an adversary cannot learn a model that works better than a random guess. In other words, we do not need to add noise as the removal of sensitive features seems to be sufficient for this data set.

2.7 Report on practical work

2.7.1 Cleaning the data set

The Kaggle data set was cleaned prior to data synthesis. More precisely, some variables were removed as they were deemed not informative enough or redundant. Also we recoded some categories and some missing information. Here is the complete list of what was carried out.

- SK_ID_CURR : Remove from data as it simply assigns a number to each observation.
- CODE_GENDER : Recode XNA values as NA.
- NAME_TYPE_SUITE : Recode empty string to NA.
- NAME_TYPE_SUITE : Combine Other_A and Other_B.
- DAYS_EMPLOYED : Recode all people employed for 1000 years to NA (this corresponds almost exclusively to retired people for which this variable does not apply).
- FLAG_MOBIL : Remove from data set, only one person did not have a mobile.
- OCCUPATION_TYPE : Recode empty string to NA.
- Variables on building in which client lives : There are three versions of each of 14 such variables : _AVG, _MODE and _MEDI. We do not understand the difference between these three types of variables and it is hard to synthesize so many variables. Since there are also five other variables with only _MODE, we keep only those.

- Document variables : There are 20 document variables, indicating whether or not someone provided a certain document. In some cases very few people provided a document, making these variables hard to synthesize. We merge all the variables in which fewer than 1000 people provided the document into a new variable.

The cleaned data set contains 82 variables for 307,511 individuals.

2.7.2 Tests with synthpop package

As mentioned previously, the synthpop package in R is designed to generate synthetic data sets for privacy protection. One of us attempted to run the main function `syn` to create a synthetic version of the cleaned Kaggle data set. The data set, however, was too large for the software. The problem is mostly the number of variables, not the number of individuals, as synthpop has been used in the past to generate synthetic versions for large data sets, for example in section 4 of this report with guidelines for generating synthetic data: https://arxiv.org/pdf/1712.04078.pdf.

It may still be possible to use this package to synthesize the cleaned Kaggle data set in a better computing environment and/or by modifying further the data set and/or modifying the specific details of the synthesizing process carried out in `syn`.

Bibliography

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pages 308–318, 2016.

[2] Hiralal Agrawal, Munir Cochinwala, and Joseph R Horgan. Automated Determination of Quasi-Identifiers Using Program Analysis, February 25 2014. US Patent 8,661,423.

[3] Luca Arciero, Claudia Biancotti, L. D'Aurizio, and C. Impenna. Exploring Agent-Based Methods for the Analysis of Payment Systems: A Crisis Model for StarLogo TNG. Journal of Artificial Societies and Social Simulation, 12(1). 2009. DOI:10.2139/ssrn.1290520

[4] Samuel Assefa, Danial Dervovic, Mahmoud Mahfouz, Tucker Balch, Prashant Reddy, and Manuela Veloso. Generating Synthetic Data in Finance: Opportunities, Challenges and Pitfalls. In 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Workshop on AI in Financial Services: Data, Fairness, Explainability, Trustworthiness, and Privacy. Vancouver, Canada, 2019.

[5] Ho Bae, Dahuin Jung, Hyun-Soo Choi, and Sungroh Yoon. AnomiGAN: Generative Adversarial Networks for Anonymizing Private Medical Data. Pac Symp Biocomput. 2020;25:563–574. PMID:31797628.

[6] Claire McKay Bowen and Joshua Snoke. Comparative Study of Differentially Private Synthetic Data Algorithms from the NIST PSCR Differential Privacy Synthetic Data Challenge. Journal of Privacy and Confidentiality, Vol. 11(1), 2021.

[7] Justin Brickell and Vitaly Shmatikov. The Cost of Privacy: Destruction of Data-Mining Utility in Anonymized Data Publishing. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 70–78, 2008.

[8] Gregory Caiola and Jerome P Reiter. Random Forests for Generating Partially Synthetic, Categorical Data. In Transactions on Data Privacy. 3, 27–42, 2010.

[9] Ramiro Camino, Christian Hammerschmidt, and Radu State. Generating Multi-Categorical Samples with Generative Adversarial Networks. International Conference on Machine Learning (ICML), Workshop on Theoretical Foundations and Applications of Deep Generative Models, 2018.

[10] Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F Stewart, and Jimeng Sun. Generating Multi-Label Discrete Patient Records Using Generative Adversarial Networks. Machine Learning for Healthcare Conference, 2017.

[11] Jörg Drechsler. Synthetic Datasets for Statistical Disclosure Control: Theory and Implementation, volume 201. Springer Science & Business Media, 2011.

[12] M. Galbiati and Kimmo Soramäki. An Agent-Based Model of Payment Systems. Journal of Economic Dynamics and Control, 35:859–875, 2011.

[13] Andre Goncalves, Priyadip Ray, Braden Soper, Jennifer Stevens, Linda Coyle, and Ana Paula Sales. Generation and Evaluation of Synthetic Patient Data. BMC Medical Research Methodology, 20:1–40, 2020.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In Advances in Neural Information Processing Systems, pages 2672–2680, 2014.

[15] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. arXiv preprint arXiv:1412.6572, 2014. Published as a conference paper at ICLR 2015. 16

[16] Volker Grimm, Uta Berger, Finn Bastiansen, Sigrunn Eliassen, Vincent Ginot, Jarl Giske, John Goss-Custard, Tamara Grand, Simone K. Heinz, Geir Huse, Andreas Huth, Jane U. Jepsen, Christian Jørgensen, Wolf M. Mooij, Birgit Müller, Guy Pe'er, Cyril Piou, Steven F. Railsback, Andrew M. Robbins, Martha M. Robbins, Eva Rossmanith, Nadja Rüger, Espen Strand, Sami Souissi, Richard A. Stillman, Rune Vabø, Ute Visser, and Donald L. DeAngelis. A Standard Protocol for Describing Individual-Based and Agent-Based Models. Ecological Modelling, 198(1):115–126, 2006.

[17] Amir Harel, Asaf Shabtai, Lior Rokach, and Yuval Elovici. M-score: Estimating the Potential Damage of Data Leakage Incident by Assigning Misuseability Weight. In Proceedings of the 2010 ACM Workshop on Insider Threats, pages 13–20, 2010.

[18] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In Advances in Neural Information Processing Systems, pages 6626–6637, 2017.

[19] Jingchen Hu. Bayesian Estimation of Attribute and Identification Disclosure Risks in Synthetic Data. arXiv:1804.02784 [stat], December 2018.

[20] Peter Kairouz, Jiachun Liao, Chong Huang, and Lalitha Sankar. Censored and Fair Universal Representations Using Generative Adversarial Models. arXiv:1910.00411, April 2021.

[21] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. arXiv:1312.6114, 2013. Published at ICLR Conference, December 2014.

[22] Jiachun Liao, Chong Huang, Peter Kairouz, and Lalitha Sankar. Learning Generative Adversarial Representations (GAP) Under Fairness and Censoring Constraints. arXiv:1910.00411, 2019.

[23] Edgar Alonso Lopez-Rojas and Stefan Axelsson. Using the RetSim Fraud Simulation Tool to Set Thresholds for Triage of Retail Fraud. In SECURE IT SYSTEMS, NORDSEC 2015, volume 9417 of Lecture Notes in Computer Science, pages 156–171, 2015.

[24] Edgar Alonso Lopez-Rojas, Dan Gorton, and Stefan Axelsson. RETSIM: A Shoe Store Agent-Based Simulation for Fraud Detection. In 25th European Modeling and Simulation Symposium, EMSS 2013, 2013.

[25] Maxim Maximov, Ismail Elezi, and Laura Leal-Taixé. CIAGAN: Conditional Identity Anonymization Generative Adversarial Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5447–5456, 2020.

[26] Casey Meehan, Kamalika Chaudhuri, and Sanjoy Dasgupta. A Non-Parametric Test to Detect Data-Copying in Generative Models. arXiv:2004.05675 [cs, stat], April 2020.

[27] Rajeev Motwani and Ying Xu. Efficient Algorithms for Masking and Finding Quasi-Identifiers. In Proceedings of the Conference on Very Large Data Bases (VLDB), pages 83–93, 2007.

[28] Vaishnavh Nagarajan, Colin Raffel, and Ian J Goodfellow. Theoretical Insights into Memorization in GANs.

[29] Jordi Nin, Javier Herranz, and Vicenç Torra. Using Classification Methods to Evaluate Attribute Disclosure Risk. In Vicenç Torra, Yasuo Narukawa, and Marc Daumas, editors, Modeling Decisions for Artificial Intelligence, Lecture Notes in Computer Science, pages 277–286, Berlin, Heidelberg, Springer, 2010.

[30] Adityanarayanan Radhakrishnan, Karren Yang, Mikhail Belkin, and Caroline Uhler. Memorization in Overparameterized Autoencoders. arXiv:1810.10333, September 2019.

[31] Jerome P. Reiter, Quanli Wang, and Biyuan Zhang. Bayesian Estimation of Disclosure Risks for Multiply Imputed, Synthetic Data. Journal of Privacy and Confidentiality, 6(1), June 2014. https://doi.org/10.29012/jpc.v6i1.635

[32] Donald B Rubin. Multiple Imputation for Nonresponse in Surveys John Wiley & Sons, 1987.

[33] Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing Generative Models via Precision and Recall. In Advances in Neural Information Processing Systems, pages 5228–5237, 2018.

[34] Asaf Shabtai, Yuval Elovici, and Lior Rokach. A Survey of Data Leakage Detection and Prevention Solutions. Springer Science & Business Media, 2012.

[35] Uthaipon Tantipongpipat, Chris Waites, Digvijay Boob, Amaresh Ankit Siva, and Rachel Cummings. Differentially Private Synthetic Mixed-Type Data Generation for Unsupervised Learning. arXiv:1912.03250, 2019.

[36] Jennifer Taub, Mark Elliot, Maria Pampaka, and Duncan Smith. Differential Correct Attribution Probability for Synthetic Data: An Exploration. In International Conference on Privacy in Statistical Databases, pages 122–137, Springer, 2018.

[37] Ke Wang and Benjamin CM Fung. Anonymizing Sequential Releases. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 414–423, 2006.

[38] Jim Young, Patrick Graham, and Richard Penny. Using Bayesian Networks to Create Synthetic Data. Journal of Official Statistics, 25(4), pp. 549–567, 2009.

[39] Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. PrivBayes: Private Data Release via Bayesian Networks. ACM Transactions on Database Systems, 42(4):1–41, November 2017.

3 Predicting the hourly Ontario energy price in the medium and long term

Soheila Samiee [d†]
Pan Liu [c†]
Arka Mukherjee [b†]
Cédric Poutré [a†]
Rémi Galarneau-Vincent [c†]
Prabodh Wankhede [b†]
Xingwei Yang [f†]
Abdoul Haki Maoude [a‡]
Andrew Day [h‡]
Ismael Assani [a‡]
Jingjing Zhang [c‡]
Maxence Prémont [c‡]
Gita Gonoody [c‡]
Mozhgan Saeidi [g‡]
Qi Guo [i‡]
Dr. Huang Huaxiong [j⋆]
Dr. Yi Yang [e⋆]

[a] *Département de mathématiques et statistique, Université de Montréal, Montréal (Québec), Canada*

[b] *Département d'informatique et de recherche opérationnelle, Université de Montréal, Montréal (Québec), Canada*

[c] *HEC Montréal, Montréal (Québec), Canada*

[d] *Mila, McGill University, Montréal (Québec), Canada*

[e] *Department of Mathematics and Statistics, McGill University, Montréal (Québec), Canada*

[f] *Smith School of Business, Queen's University, Kingston (Ontario), Canada*

[g] *Department of Computer Science, Dalhousie University, Halifax (Nova Scotia), Canada*

[h] *Applied Mathematics, Western University, London (Ontario), Canada*

[i] *Department of Mathematics and Statistics, University of Calgary, Calgary (Alberta), Canada*

† *Machine learning group*

‡ *Modeling group*

⋆ *Supervisor*

September 2021
Les Cahiers du GERAD

3.1 Introduction

The sales planning team at Hydro-Québec needs a forecast of electricity spot prices to produce a forecast of total sales and the resulting revenue, in the medium and long term. Forecasting spot prices on the Ontario market (an important one for Hydro-Québec) is especially difficult. [1, 2, 3, 4, 5].[1] There are three main reasons for this.

- There are many fixed-price supply contracts.
- A fairly large portion (12%) of the supply is coming from wind-based resources.
- There is a lot of uncertainty in the demand.

3.2 Our goals

Our goal is to forecast the Ontario Energy Price for medium-term and long-term periods. Below we describe the data set and outline approaches to this problem. Samiee, Liu, Mukherjee, Poutré, Galarneau-Vincent, Wankhede, and X. Yang worked on machine learning; Haki Maoude, Day, Assani, Zhang, Prémont, Gonoody, Saeidi, and Guo worked on modelling. The supervisors were Dr. Huang Huaxiong and Yi Yang.

3.3 Data set

Available data consisted of:

- Predicted weekly data (18-month predictions): 2015–2020;
- Historical hourly data: 2017–2020.

Three 18-months predictions data files were selected by Hydro-Québec partners as test data for comparing the performance of each suggested algorithm with the current company benchmark.

3.4 Machine learning approach

In this section the machine learning methods used for price prediction are briefly introduced and the results are illustrated. The price regression is carried out in two steps: Feature extraction, and then Regression on validation data. Then the best regression method based on validation data is used for price regression on the test data. More details are provided in the following.

3.4.1 Data

Predicted weekly data (excluding three test files) were used for training and validation of the algorithms with 75% and 25% ratios, respectively.

3.4.2 Feature extraction

There were 40 predicted features available for each week in the training data, including but not limited to *expected total and peak energy demand of different regions of Ontario*, and *average and peak temperature in that week*. Not all of the provided features, however, are useful in energy price prediction. Therefore more relevant features were selected with the three following approaches. (1) Linear correlation of each

[1] http://www.ieso.ca/en/Power-Data/Price-Overview/Hourly-Ontario-Energy-Price

feature with the real energy price estimated using Pearson Correlation: 10 features with the highest correlations were selected (Figure 3.1). (2) Mutual information between each feature and the price: 10 features with the highest correlations were selected (Figure 3.2). (3) Features with non-zero coefficient from Lasso regression with α equal to 0.4. These selected features were concatenated to form the best group of features, which were then used in the regression. A total of 24 features were selected through this process. These features are shown in Figure 3.3.

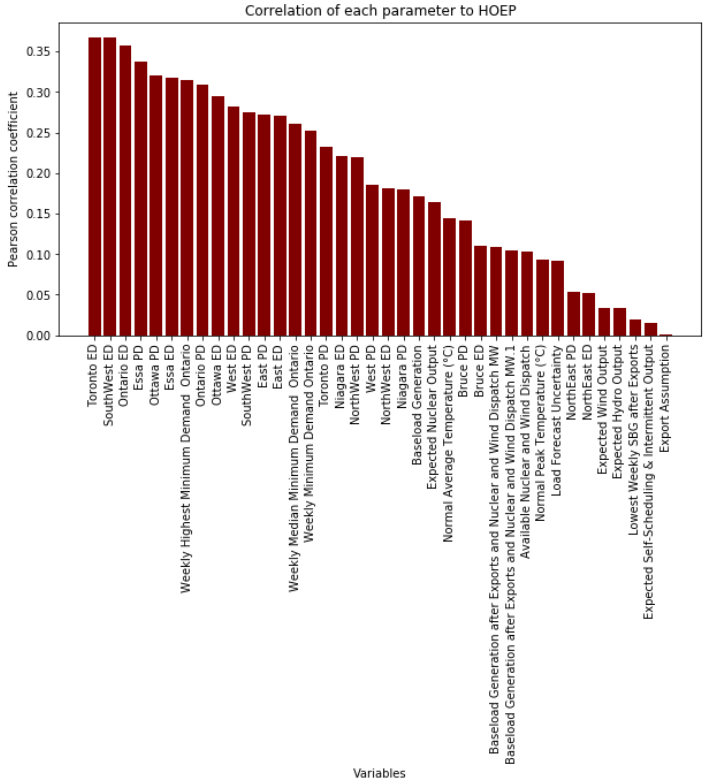

Figure 3.1: Sorted features based on their correlation with the real price.

3.4.3 Regression

Five different algorithms, including *Linear Regression*, *Lasso Regression*, *Elastic Net*, *Neural Network (Multi Layer Perceptron)*, and *Gradient Boosting* (GB), were used for regressing the price based on selected features. All parameters were set using a portion of the validation data set (further details on hyper parameters and their tuning are available in the Python code). Table 3.1 shows the root-mean-square error (rMSE) in price estimation for each algorithm. On validation data the Gradient Boosting approach had the best performance.

Table 3.1: Validation error for different regression algorithms.

	Linear Regression	Lasso	Elastic Net	Neural Network	Gradient boosting
rMSE (price)	7.95	7.98	7.93	8.91	**5.15**

The sorted selected features based on their importance in the regression with Gradient Boosting are illustrated in Figure 3.3.

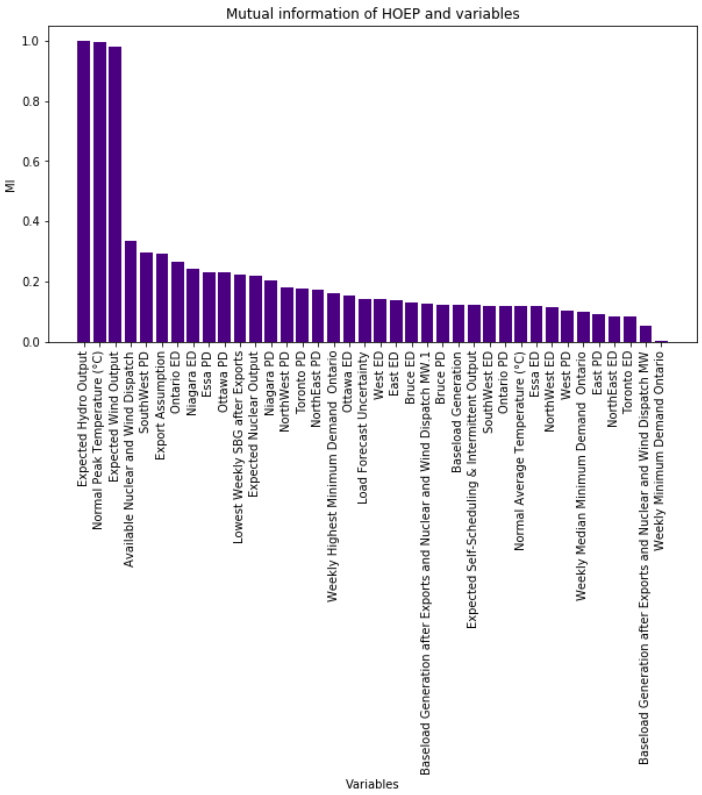

Figure 3.2: Sorted features based on their mutual information with the real price.

3.4.4 Results

As already explained the goal was to obtain, for the Ontario energy price, a better forecast than the available Hydro-Québec benchmark. Based on validation results Gradient Boosting is the regression method with the best performance. Therefore this algorithm was used for predicting the price in test data files. Furthermore a price forecast with linear regression (as a vanilla regression method) was also computed. Figure 3.4 illustrates the real price (black trace), the benchmark (green trace), and the price predicted by the Gradient boosting regression (red trace) for all three 18–month prediction test files.

For a quantitative comparison the rMSE of results for benchmark, Gradient Boosting regression, and Linear regression for all three files were computed (Table 3.2). Figure 3.5 summarize these results, and shows that regression with Gradient Boosting could outperform the benchmark and linear regression in all three test files.

Table 3.2: Test error for benchmark, Gradient Boosting, and linear regression algorithms.

Test file ♯	Prediction date	Benchmark	Linear regression	Gradient boosting
1	March 2015	12.85	6.71	**4.38**
2	March 2018	7.27	8.04	**6.33**
3	September 2018	9.6	6.25	**4.65**

3.5 Modelling approach

In this section we propose some models that can help gain a better understanding of the dynamic of the HOEP.

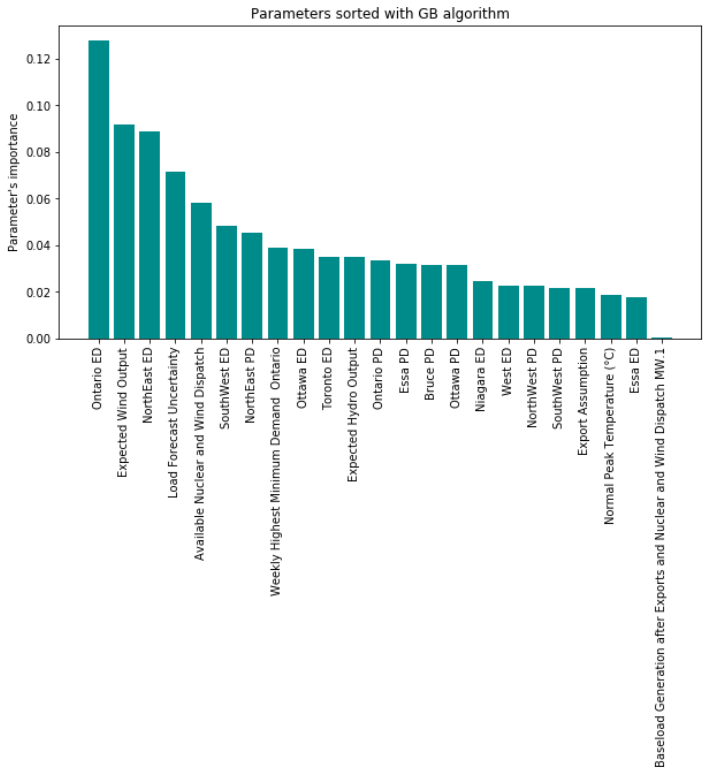

Figure 3.3: All 24 features selected based on Pearson Correlation, Mutual Information, and Lasso Regression, after sorting based on their importance in the Regression with Gradient Boosting.

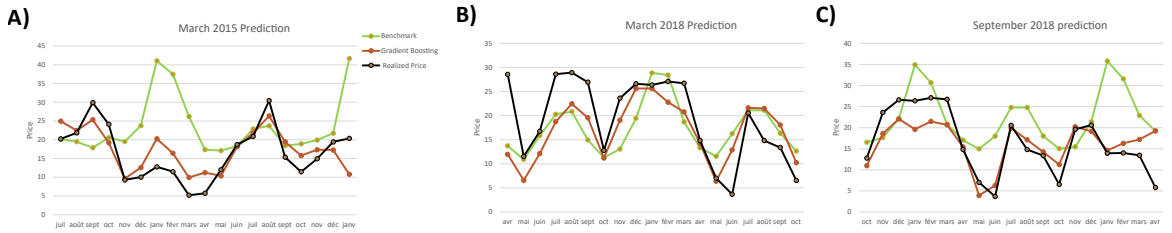

Figure 3.4: Test results: Comparison of predicted prices for the next 18 months using our Gradient Boosting regression (red) with the benchmark (green) and real price (black) for three test files.

3.5.1 Daily historical regression

Model

The following model is trained on the daily historical data. This model proposes a linear relationship between several variables for which we have the forecasts for the relevant time horizon. Also as some of those forecast variables are weekly and we want to use a daily model, we suppose the model takes into account two types of explanatory variables. The model is as follows:

$$HOEP_t = X_t\beta_1 + Z_t\beta_2 + \epsilon_t\,, \tag{3.1}$$

where X_t is the set of variables for which only weekly forecasts are available and Z_t is the set of variables for which daily forecasts are available. Here t represents the day and ϵ_t is a normal random variable with mean 0 and variance $\sigma^2 > 0$.

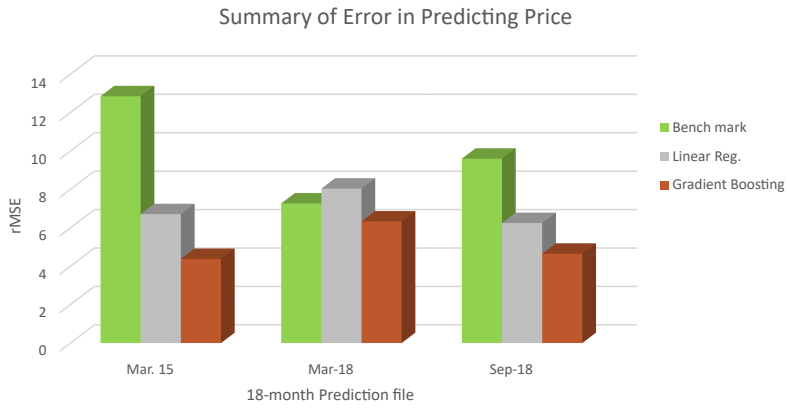

Figure 3.5: rMSE error for predicted price in benchmark and with Linear regression and Gradient Boosting algorithms.

Based on this model, the prevision of $HOEP$ for a specific horizon h from a specific date t is as follows.

$$\widehat{HOEP}_{t+h|t} = \mathbb{E}\left[HOEP_{t+h}|\mathcal{I}_t\right] = \mathbb{E}\left[X_{t+h}\hat{\beta}_1 + Z_{t+h}\hat{\beta}_2 + \hat{\epsilon}_{t+h}|\mathcal{I}_t\right], \qquad \forall\, t, h$$
$$\widehat{HOEP}_{t+h|t} = \mathbb{E}\left[X_{t+h}|\mathcal{I}_t\right]\mathbb{E}\left[\hat{\beta}_1|\mathcal{I}_t\right] + \mathbb{E}\left[Z_{t+h}|\mathcal{I}_t\right]\mathbb{E}\left[\hat{\beta}_2|\mathcal{I}_t\right] + \mathbb{E}\left[\hat{\epsilon}_{t+h}|\mathcal{I}_t\right]$$

We assume that explanatory variables (and their forecast) are exogenous. Then $X_{t+h|t} := \mathbb{E}\left[X_{t+h}|\mathcal{I}_t\right]$ and $Z_{t+h|t} := \mathbb{E}\left[Z_{t+h}|\mathcal{I}_t\right]$ are assumed to be known. Therefore we have

$$\widehat{HOEP}_{t+h|t} = X_{t+h|t}\mathbb{E}\left[\hat{\beta}_1|\mathcal{I}_t\right] + Z_{t+h|t}\mathbb{E}\left[\hat{\beta}_2|\mathcal{I}_t\right]. \tag{3.2}$$

In reality only weekly forecasts of the variables in vector X are known. Fortunately we only need weekly forecasts of the HOEP. Then we compute the weekly forecast of the HOEP by averaging the daily forecasts. We then obtain

$$\widehat{HOEP}_{t+h:(t+h+6)|t} := \frac{1}{7}\sum_{j=h}^{h+6}\widehat{HOEP}_{t+j|t} \qquad \forall\, h = 1, 8, 15, \ldots.$$
$$\widehat{HOEP}_{(t+h):(t+h+6)|t} = \left(\frac{1}{7}\sum_{j=h}^{h+6}X_{t+j|t}\right)\mathbb{E}\left[\hat{\beta}_1|\mathcal{I}_t\right] + \left(\frac{1}{7}\sum_{j=h}^{h+6}Z_{t+j|t}\right)\mathbb{E}\left[\hat{\beta}_2|\mathcal{I}_t\right].$$

Not every term $X_{t+j|t}$ is known but the average $X_{(t+h):(t+h+6)|t} := \left(\frac{1}{7}\sum_{j=h}^{h+6}X_{t+j|t}\right)$ is known. Then the following holds.

$$\widehat{HOEP}_{(t+h):(t+h+6)|t} = X_{(t+h):(t+h+6)|t}\mathbb{E}\left[\hat{\beta}_1|\mathcal{I}_t\right] + \left(\frac{1}{7}\sum_{j=h}^{h+6}Z_{t+j|t}\right)\mathbb{E}\left[\hat{\beta}_2|\mathcal{I}_t\right]$$

We propose the following procedure to forecast the price.

1. Based on daily historical data, estimate the parameters β_1 and β_2 (training phase).
2. Use the weekly value $X_{(t+h):(t+h+7)|t}$ for each day of the forecast period (forecast data preparation).
3. Compute the daily forecast $\widehat{HOEP}_{t+h|t}$ of the HOEP using Equation (3.2). This value is not a good daily forecast.
4. Compute a weekly average of the predictions, denoted $\widehat{HOEP}_{t+h:(t+h+7)|t}$. This is a good weekly forecast (forecasting phase).

Estimation and results

For the estimation, we defined variable X_t as the vector whose components are the daily energy demand and the daily mean temperature, and variable Z_t as the vector whose components are the seasons (Winter, Spring, Summer, and Fall) and a component indicating whether the day is a weekend day or a vacation day. The Z_t components are almost always known years in advance and the X_t components are given by the Ontario forecast.

It is possible to choose a data set and train the model on that data set, and then to compute a forecast for another data set. Once the model has been trained, the HOEP forecast for a given week can be derived from the knowledge of the following: season, weekend/vacation days, demand forecast, and temperature.

In order to test the model, we train the model only with all the available data before the day t where the forecasting starts. The following Table 3.3 displays the data and the model performance in terms of the RMSE (for the benchmark and the regression).

Table 3.3: Daily historical regression performance.

Scenario	Training period	Forecasting period	RMSE	
			Benchmark	Regression
March 2018	01-06-2015 to 31-03-2018	01-04-2018 to 30-09-2019	7.33	8.52
September 2018	01-06-2015 to 30-09-2018	01-10-2018 to 31-03-2020	9.34	5.34

The following two graphs present the forecast for the two scenarios displayed in the table.

3.5.2 Weekly provisional regression

First, as Hydro-Québec is receiving the forecast data on a weekly basis, we decided to aggregate the hourly data to work on a weekly model.

Second, we noticed that the available observed variables are different from the available forecast variables. We decided to use in the model only the variables included in the forecast file and to exclude all other variables.

We decided to train our model on the observed data and test our model on the forecast data.

Data

Before using the stepwise selection method on all the raw variables, we decided to take a close look at the bivariate relation between HOEP and some of the variables that could have a big impact on the model.

We took a close look at the bivariate relation between HOEP and the temperature. From our point of view, the relation can be split into different parts. We tested different scenarios for the relationship between the price and the temperature. We let the selection method decide which one to choose in the end.

We also consider the square of the temperature as a possible variable of the model.

Model

We trained our model on the period from January 8^{th}, 2017 to March 31^{th}, 2019; i.e., there were 117 observations in our training sample.

Figure 3.6: March 2018 and September 2018 forecasts : Realized HOEP in black, Benchmark in green, and the model forecasts in orange.

To model the HOEP we used a linear regression. To select the more useful variables to include into the model, we used the iterative stepwise selection method. The final model includes the following variables.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-23.8578	18.9616	-1.26	0.2113
NUCLEAR_mean	-0.0069	0.0008	-9.14	0.0000
WIND_mean	-0.0065	0.0015	-4.31	0.0000
Temp (C)_mean	-0.4025	0.2025	-1.99	0.0496
Northwest_max	0.0436	0.0321	1.36	0.1776
East_max	0.0129	0.0061	2.10	0.0379
Toronto_max	-0.0029	0.0015	-2.02	0.0463
Northeast_sum	0.0001	0.0001	1.57	0.1189
Northwest_sum	-0.0005	0.0002	-2.37	0.0195
Ottawa_sum	0.0002	0.0001	3.83	0.0002
Bruce_sum	0.0003	0.0001	2.32	0.0223
Southwest_sum	0.0002	0.0000	5.53	0.0000
HYDRO_min	-0.0086	0.0013	-6.60	0.0000
Temp (C)_mean2	-0.0931	0.0268	-3.47	0.0008
meanTempLT11	-2.9779	0.6363	-4.68	0.0000
meanTempLT11_2	0.2429	0.0569	4.27	0.0000
meanTempLTm5	2.0785	0.8483	2.45	0.0160
meanTempLT57	1.5027	0.3920	3.83	0.0002

Here are the transformed variables that were selected by the stepwise selection procedure.

- meanTempLT11: the average temperature if the average temperature is lower than or equal to 11 degrees, otherwise 0.
- meanTempLT11_2: the square of meanTempLT11.

- meanTempLTm5: the average temperature if the average temperature is lower than minus 5 degrees, otherwise 0.
- meanTempLT57: the average temperature if the average temperature is greater than or equal to minus 5 degrees and lower than 7 degrees, otherwise 0.
- Temp (C)_mean2: the square of the average temperature.

Results

In Table 3.4, we give the root-mean-square error (rMSE) that we obtained for the three reference periods. As we can see, on the first two reference periods, the performance of our model is worst than the benchmark but on the last reference period, our model performs better than the Hydro benchmark in terms of rMSE. Maybe the performance would have been better if we had trained the model on the full set of observations instead of using only the data until March 2019. It is possible to do that as we use the forecast data to test our model.

Table 3.4: Comparison of the root-mean-square error (rMSE) of our model with that of the Hydro benchmark

Reference period	Benchmark	Linear regression
March 2015	13.93	16.85
March 2018	7.27	8.96
September 2018	9.6	9.55

Figure 3.7 displays our forecasts (orange line) versus the realized price (black line) and the Hydro benchmark (blue line).

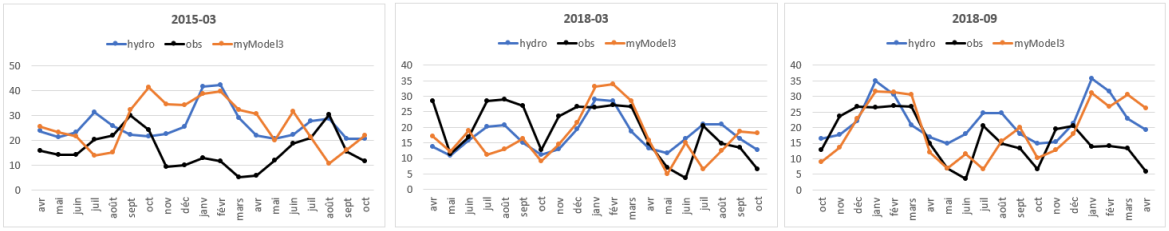

Figure 3.7: March 2018 and September 2018 forecasts : Realized HOEP in black, Benchmark in green, and the model forecasts in orange.

It would be interesting to use the same approach but instead of training the model on the aggregated observed data, to select the variables directly and train the model on the forecast data. This approach would certainly yield better results as the distribution of the test sample would be closer to the distribution of the values in the training sample.

3.6 Conclusions

As can be seen in Figure 3.8, which compares different approaches, the Gradient Boosting method of machine learning yields the best root-mean-square error for all different time periods; it outperforms the benchmark.

Bibliography

[1] CL Anderson and M Davison. A Hybrid System-Econometric Model for Electricity Spot Prices: Considering Spike Sensitivity to Forced Outage Distributions. IEEE Transactions on Power Systems, 23(3):927–937, 2008.

RMSE of Error

		Classical Machine Learning			Deep Learning		Modeling			
	Benchmark	Linear Reg.	Gradient Boosting	NL Model	LSTM (weekly +PCA)	LSTM (Hourly)	TS 1 month	TS 3 month	TS monthly	ARX with 3 features
Mar. 2015	12.85	6.71	**4.38**	9.1						
Mar. 2018	7.27	8.04	**6.33**	8.49	7.27	7.45				12.8673
Sep. 2018	9.6	6.25	**4.65**	5.55	7.49	7.9			8.03	7.5
Dec.18	14.4			5.18			**4.88**	6.57		7.1882

Figure 3.8: Comparison of RMSE for different Approaches

[2] Manuela Buzoianu, Anthony Brockwell, and Duane Seppi. A Dynamic Supply-Demand Model for Electricity Prices. Carnegie Mellon University. Journal contribution. https://doi.org/10.1184/R1/6586331.v1 2005.

[3] Jessica Hwang, Paulo Orenstein, Judah Cohen, Karl Pfeiffer, and Lester Mackey. Improving Subseasonal Forecasting in the Western US with Machine Learning. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2325–2335, 2019.

[4] IESO. http://www.ieso.ca/en/power-data/price-overview/hourly-ontario-energy-price.

[5] Jiahan Li and Ilias Tsiakas. Equity Premium Prediction: The Role of Economic and Statistical Constraints. Journal of Financial Markets, 36:56–75, 2017.

4 Predictive risk modelling in aviation incidents

Prakash Gawas [a]
Hyuntae Jung [b]
Denis Larocque [c]
Michael R. Lindstrom [d]
Guillaume Poirier [e]
Ahmed Sid-Ali [f]

[a] *Polytechnique Montréal, Montréal (Québec), Canada*

[b] *International Air Transport Association, Montréal (Québec), Canada*

[c] *GERAD & HEC Montréal, Montréal (Québec), Canada*

[d] *University of California, Los Angeles, Montréal (Québec), Canada*

[e] *IVADO, Montréal (Québec), Canada*

[f] *Carleton University, Montréal (Québec), Canada*

September 2021
Les Cahiers du GERAD

Abstract: *In the context of the Tenth Montréal Industrial Problem Solving Workshop, the International Air Transport Association posed a challenge to participants: to identify anomalies in time series data for flights, across different aircraft types and airport origins/destinations. Within this anomaly detection problem two questions arise: how to identify a time series as anomalous and how to identify when a new record is anomalous relative to previous data in the time series. We present our analysis and a novel method of time series anomaly detection using an extension of kernel density estimation.*

4.1 Introduction

Currently the global aviation safety risk identification is mainly reactive, i.e., its approach is that "we don't know what can be the problem until we face the problem." The International Air Transport Association (IATA) is interested in proactively identifying potential risk areas before they evolve into an accident. Thus we need to look at the data to search for "hints" about where to focus. On a global scale, collecting, processing, and analyzing these data sets manually are unsustainable. We need the support of automation for monitoring the risk area continuously.

In this report we focus upon the two problems outlined below. Some members of our team have also published an article on the novel method developed for Problem 1 and we refer the reader to this article for a broader, more technical exposition [1].

4.1.1 IPSW challenge target 1: Anomaly detection

The goal is to develop a model to give hints to safety analysts on where to look, instead of them having to query every criterion one by one. The model should examine the set of incident reports by, for example, drilling down into specific aircraft types, concluding (for instance) that

- Aircraft Type A reports are not significantly different from the global rate, or
- Aircraft Type B reports display an anomalous behaviour relative to the global rate, which may indicate a prominent safety risk.

Once the model automatically identifies such "anomalies" with statistical evidence, a flag will be raised, so that human safety analysts can carry out a deeper investigation.

4.1.2 IPSW challenge target 2: Predictive analysis

Here the goal is to develop a model to predict event rates based on historical records, and raise a flag if the actual rate is exceptional. For example suppose we are given monthly rates for Event A (with the seasonal pattern). After training with, say, two years of historical incident data, the model should make a prediction for the next month with a given interval of confidence. The actual data for the next month, however, may be out of the bounds. Then this data should be flagged as anomalous.

4.1.3 Data – Incident reports & sector

We were provided with Incident Reports: approximately $621,000$ reports including many details. For example one report could include the following items.

- Report ID: 7723515
- Year: 2018
- Month: May

- Fleet Family: ACType5
- Location: Airport162
- Location Country: Country256
- Phase: Approach
- Event: Weather – Windshear

We were also provided with Sector Data, to normalize the flights by the number of flights between a given source and a given destination over a given time window. The data were provided on a quarterly basis. Here is an example.

- Quarter: 2018 Q2
- Fleet Family: ACType5
- Departure: Airport162
- Departure Country: Country256
- Arrival: Airport359
- Arrival Country: Country26
- Sectors: 3,631.

This allows us to compute the flight statistics on a per 1000 flight basis, for instance.

4.2 Problem solving

We present a series of ideas that could be used in studying anomalies.

- Vectorized representation for data and Logistic Regression
- Neural Networks
- Naive Bayes Classifiers
- Functional KDE
- Functional Isolation Forest
- Time-series Forecasting (e.g. the Forecast and Prophet `R` packages)

4.2.1 Data preparation

As a preliminary work, we wrote scripts to process the raw data into a form that could be analyzed for anomaly detection. The scripts allowed a user to specify certain descriptors of the events they are looking for and then to obtain time series for those events by fleet or location. For example a user could obtain the time series for all aircraft types for records that listed both “Windshear” and “Turbulence”.

4.2.2 Anomaly detection

We used two methods for anomaly detection: in the first method, we extended Kernel Density Estimation in a novel fashion to assign a score to the time series for its level of anomalousness; in the second method we used Hierarchical Curve Clustering with the dtwclust `R` package.

Functional KDE anomaly detection

Our thought process in developing an extension of KDE for time series can be summarized as follows.

- Think of an anomaly as being distant from the rest of the data.
- If the data comes from some distribution, anomalies should have correspondingly small “probability densities.”

- Using our data (a collection of time series), we want to ascribe a score to represent these densities so that comparatively low scores represent anomalies.
- Since we don't know the distribution we use Kernel Density Estimation.

Kernel Density Estimation (KDE) review. Kernel Density Estimation (KDE) uses sums of Gaussian kernels to infer empirical, continuous probability distributions for data. Consider discrete samples of a Weibull distribution with probability density function (pdf)

$$f(x) = kx^{k-1}e^{-x^k} \quad \text{for} \quad k = 2. \tag{4.1}$$

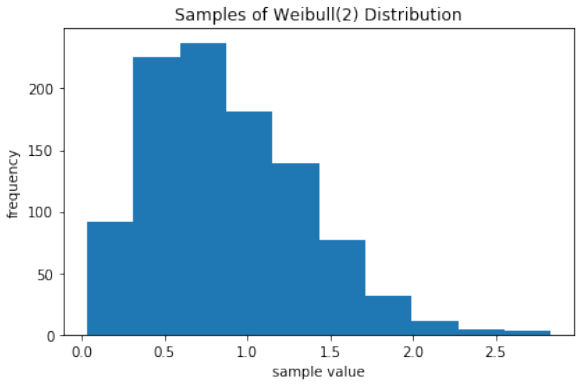

Figure 4.1: Sample of Weibull distribution

If we place a Gaussian kernel at each point, then the sum of all such kernels yields an estimate for the true pdf with lower pdf values indicating anomalies – see Figure 4.1. We also remark that values whose probability density is very low tend to be anomalous as depicted in Figure 4.2. Thus if we could ascribe "probability densities" to time series, which are points in a Hilbert space, then we could likewise identify anomalous time series as depicted in Figure 4.3.

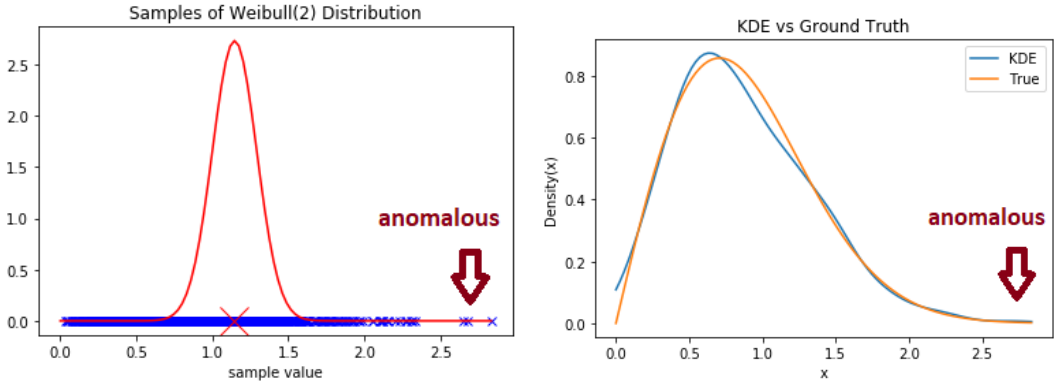

Figure 4.2: Intuition of anomalies being points far away from the peak density values.

Simple functional KDE. In our first approach, we can think of our time series as samples of signals $x : [0, T] \to \mathbb{R}$ or as being in the Hilbert space, $\mathcal{H}$, say $L^2(0, T)$ or $\mathcal{H}^1(0, T)$. Hilbert spaces have induced norms, $||\cdot||$, which can be thought of as generalized distances. The idea is to place a Gaussian kernel over $\mathcal{H}$ at each time series $x_i(t)$ and construct a probability density functional.

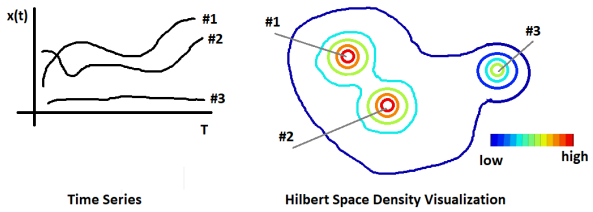

Figure 4.3: Two curves are very close and there is one anomaly. Then that curve, when abstractly mapped to a probability density, has a lower probability density in its vicinity.

We can formally, without rigor, define an empirical pdf over $\mathcal{H}$ as follows.

- Begin with a sample of curves $S = \{x_j(t), j = 1, \ldots, N\}$, where $x_j \in \mathcal{H}$ for $j = 1, 2, \ldots N$.
- Choose $\sigma > 0$, a hyper-parameter.
- Define the probability density functional

$$\rho(a) = \sum_{x \in S} e^{-\frac{1}{2\sigma^2}(x-a)^2}. \tag{4.2}$$

- Assign to each x_j a score $s_j = \rho[x_j]$.
- Identify anomalies by a histogram of $s_j, j = 1, \ldots, N$.

For "High-Energy/Unstable Approach," scores that are at most 10 seem anomalous (by inspection) – see Figure 4.4.

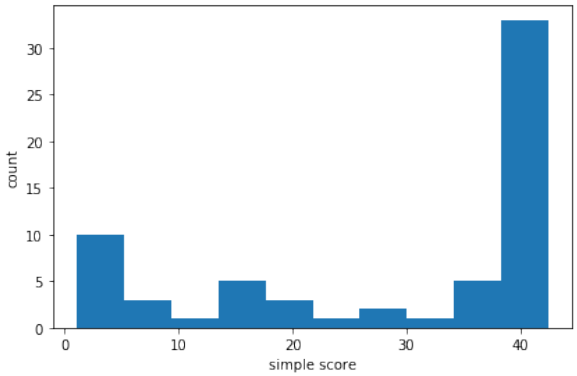

Figure 4.4: Histogram of scores using the simple approach.

Discrete Fourier Transform functional KDE

In our second approach, we note that $L^2([0,T])$ and $H^1([0,T])$ have countable bases $\{e2\pi in/T\}_{n \in Z}$. Fix M and suppose $x_j(t) \approx \sum_{n=-M}^{M} \hat{x}_n^j e^{2\pi i n/T}$. Suppose that each $\hat{x}_n \sim \epsilon_n$ for some pdf ϵ_n with corresponding density over $\mathbb{C}$ of $\zeta_n(z)$. Then to each curve x_j we can ascribe a pdf value in $\mathbb{R}^{2N+1}$ with

$$f(x_j) = \prod_{n=-M}^{M} \zeta_n(\hat{x}_n^j). \tag{4.3}$$

In practice we use a Discrete Fourier Transform (DFT) since our signal is discrete and finite. A method is summarized below.

- Begin with a set of curves $S = \{x_j(t), j = 0, \ldots, N-1\}$ where $x_j \in \mathcal{H}$ for $j = 0, 1, \ldots N-1$.
- Use a Discrete Fourier Transform to compute $\{\hat{x}_n^j | j = 0, 1, \ldots, N-1; n = 0, 1, \ldots, M-1\}$.
- Use KDE to estimate pdf of $\hat{x}_n$, call it ζ_n for $n = 0, \ldots, M-1$.
- Define the probability density at $a \in \mathcal{H}$ as

$$\rho[a] = \prod_{n=0}^{M} \zeta_n(\hat{x}_n). \tag{4.4}$$

- Assign to each x_j a score $s_j = \rho[x_j]$.
- Identify anomalies by a histogram of $s_j, j = 1, \ldots, N$.

In Figure 4.5, we display an example distribution of $\hat{x}_1$ values. KDE is carried out upon this in each Fourier mode. For "High-Energy/Unstable Approach," scores that are at most -510 are anomalous by inspection – see Figure 4.6.

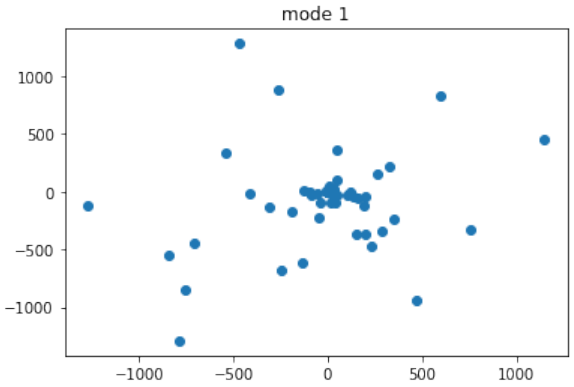

Figure 4.5: Distribution of Discrete Fourier coefficients at mode number $m = 1$.

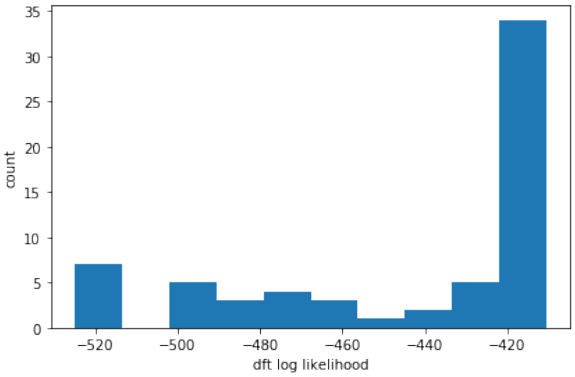

Figure 4.6: Histogram of time series scores using the Fourier approach.

Simple vs DFT comparisons. For selected events we display the anomalous aircraft numbers for the two methods in Table 4.1. There is a significant overlap between the two methods for computing anomalous flight IDs. Everything the DFT method finds is also found by the Simple method.

We plot the time series of "High Energy/Unstable Approach" in Figure 4.7; the ordinary curves are in blue, based on the DFT classification. The anomalous curves are red. Interpretation (i.e., identifying why a curve is anomalous) is an open question.

Table 4.1: Anomalous aircrafts for both methods and selected event types.

Events	Simple	DFT
Landing Gear System	6, 11, 12, 13, 14, 23, 25, 29, 33, 48, 52	11, 23, 25, 33, 48, 52
High Energy/Unstable Approach	11, 12, 13, 14, 16, 18, 19, 20, 22, 23, 30, 36, 52, 57	13, 19, 30, 36, 52, 57
Windshear	8, 9, 12, 13, 14, 16, 20, 21, 22, 26, 30, 51	8, 12, 14, 20, 26, 30, 51

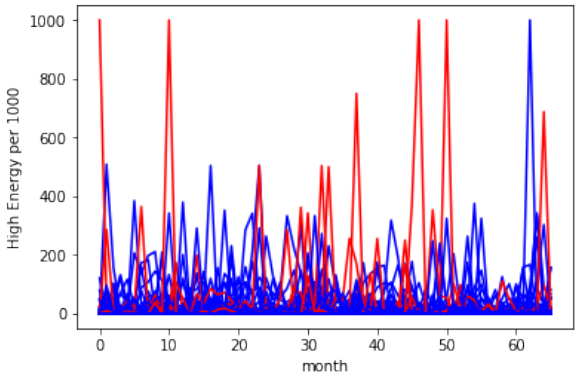

Figure 4.7: Time series of "High Energy/Unstable Approach" events with anomalous series in red.

Anomaly detection —- Hierarchical curve clustering

Predictive analysis

Here we have a classical time-series forecasting problem. Several methods are available and implemented in readily available software/languages like `R`. The following example uses a moving window scheme in order to forecast each of the last 12 months, using the previous months to fit the model with the Prophet `R` package (see Figure 4.9).

4.3 Next steps

Following the workshop there are a number of steps that could be taken to further the development of our methods and make the results more useful. We list a few of them below.

- Automate data creation and management, including the verification of data quality.
- Try and compare several anomaly detection methods to find which one has the best performance and suits IATA's needs (see the Appendix for a list of methods).
- Automate data analysis, including data extraction.
- Prepare visualization and reporting tools, dashboards, etc.
- One way to proceed would be to have an M.Sc. student from HEC Montréal do a supervised project (internship) at IATA.
- A supervised project consists of 400 hours of work within one semester (4 months).
- Students in the specializations "Business Intelligence" or "Data Science and Business Analytics" are perfectly equipped with the technical and managerial skills required for this project.

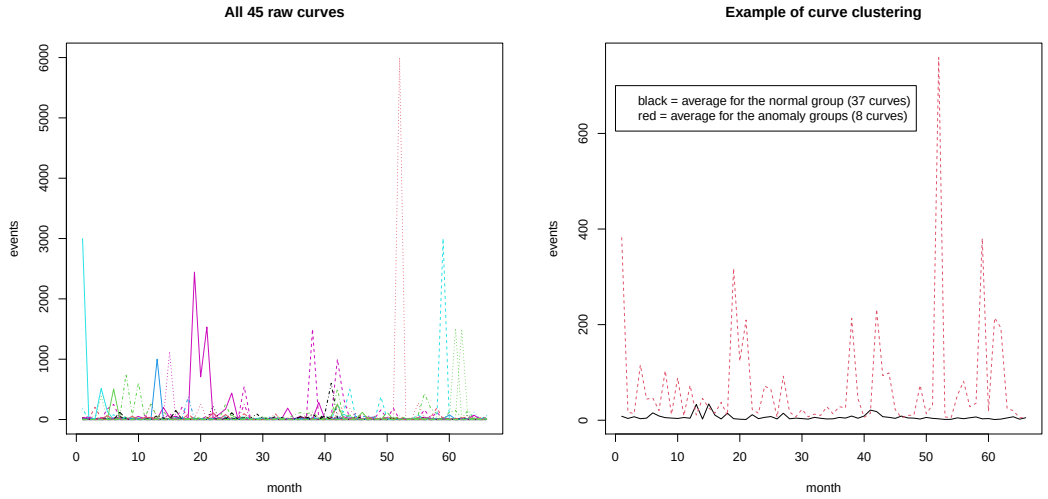

Figure 4.8: Descriptor: Windshear. Curve by Fleet Family (2013 – 2018 Aggregated).

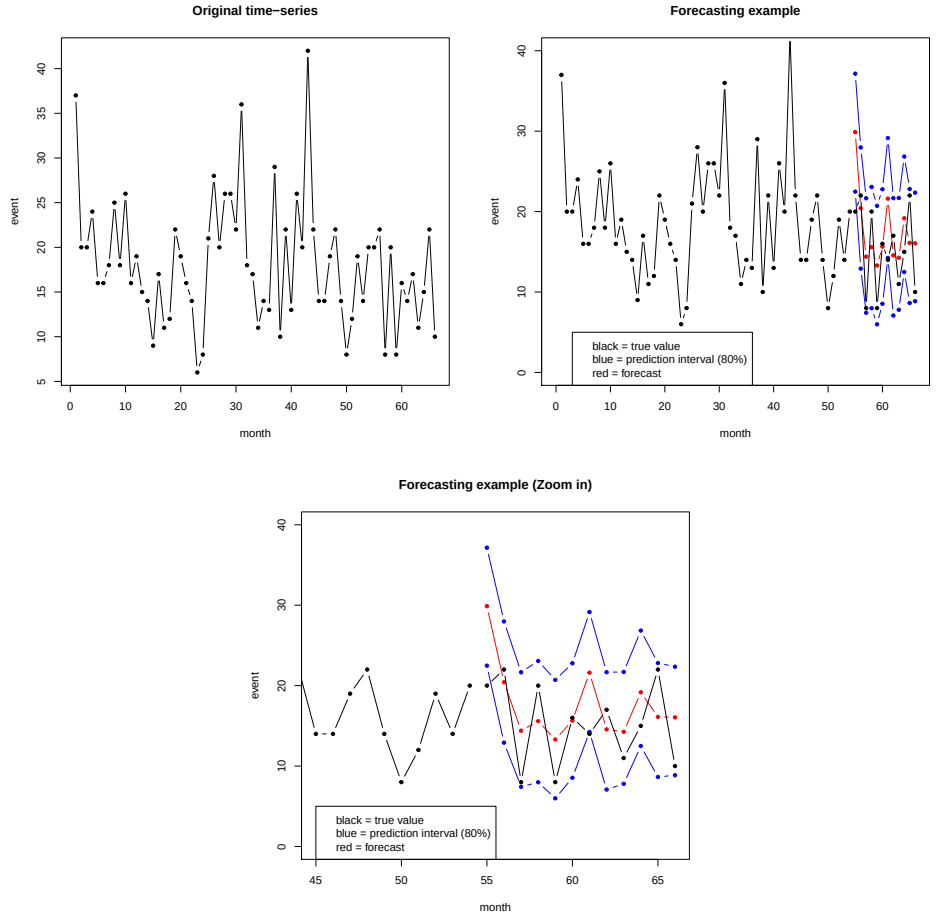

Figure 4.9: Event Type: Landing Gear. Fleet Family: Aircraft Type 1. Monthly # of events.

Acknowledgments

We would like to thank the Centre de recherches mathématiques (CRM) and the Institute for Data Valorization (IVADO) for hosting the workshop online.

Author contributions

PG helped in data processing. HJ helped by presenting the problem, providing data, and analyzing/interpreting the results. DL helped in developing and running methods and writing this report. MRL helped in data processing, developing and running methods, and writing this report. GP helped in coordinating the work. AS helped write this report.

Appendix

A1 Additional anomaly detection methods

- A few possible methods for problem 1 (anomaly detection).
  - Time-series clustering (R package dtwclust).
  - Functional isolation forest (Python code: https://github.com/Gstaerman/FIF), https://arxiv.org/abs/1904.04573.
  - Robust archetypoids (R package ada methods). https://link.springer.com/article/10.1007/s11634-020-00412-9.
  - Control chart for functional data (R package qcr). https://www.mdpi.com/1099-4300/20/1/33.
- Possible methods for problem 2 (time-series forecasting).
  - Numerous R packages available: https://cran.r-project.org/web/views/TimeSeries.html.
  - e.g.: Fable, Forecast, Prophet.

Bibliography

[1] M. R. Lindstrom, H. Jung, and D. Larocque. Functional Kernel Density Estimation: Point and Fourier Approaches to Time Series Anomaly Detection. Entropy, 22:1363, 2020.