N°d'ordre NNT : XXXXX

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
l'Institut National des Sciences Appliquées de Lyon
En cotutelle internationale avec
l'Université de Passau

Ecole Doctorale N° EDA 512
INFOMATHS

Spécialité de doctorat :
Informatique

Soumise en Septembre 2019, par :
Yvan, François, Marcel LUCAS

Credit Card Fraud Detection using Machine Learning with Integration of Contextual Knowledge

---

Devant le jury composé de :

Prof. Dr. Chantal Soulé-Dupuy (Université de Toulouse 1 Capitole)
Prof. Dr. Eric Gaussier (Université Grenoble Alpes)
Prof. Dr. Mathias Lux (Alpen-Adria Universität)
Dr. Gabriele Gianini (University of Milan)
Prof. Dr. Sylvie Calabretto (INSA Lyon)
Prof. Dr. Michael Granitzer (Universität Passau)
Dr. Pierre-Edouard Portier (INSA Lyon)
Dr. Léa Laporte (INSA Lyon)

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **http://www.edchimie-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage **secretariat@edchimie-lyon.fr** INSA : R. GOURDON	M. Stéphane DANIELE Institut de recherches sur la catalyse et l'environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 Avenue Albert EINSTEIN 69 626 Villeurbanne CEDEX **directeur@edchimie-lyon.fr**
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** **http://edeea.ec-lyon.fr** Sec. : M.C. HAVGOUDOUKIAN **ecole-doctorale.eea@ec-lyon.fr**	M. Gérard SCORLETTI École Centrale de Lyon 36 Avenue Guy DE COLLONGUE 69 134 Écully Tél : 04.72.18.60.97 Fax 04.78.43.37.17 **gerard.scorletti@ec-lyon.fr**
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** **http://e2m2.universite-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : H. CHARLES **secretariat.e2m2@univ-lyon1.fr**	M. Philippe NORMAND UMR 5557 Lab. d'Ecologie Microbienne Université Claude Bernard Lyon 1 Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69 622 Villeurbanne CEDEX **philippe.normand@univ-lyon1.fr**
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** **http://www.ediss-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : M. LAGARDE **secretariat.ediss@univ-lyon1.fr**	Mme Emmanuelle CANET-SOULAS INSERM U1060, CarMeN lab, Univ. Lyon 1 Bâtiment IMBL 11 Avenue Jean CAPELLE INSA de Lyon 69 621 Villeurbanne Tél : 04.72.68.49.09 Fax : 04.72.68.49.16 **emmanuelle.canet@univ-lyon1.fr**
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** **http://edinfomaths.universite-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 **infomaths@univ-lyon1.fr**	M. Luca ZAMBONI Bât. Braconnier 43 Boulevard du 11 novembre 1918 69 622 Villeurbanne CEDEX Tél : 04.26.23.45.52 **zamboni@maths.univ-lyon1.fr**
Matériaux	**MATÉRIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction **ed.materiaux@insa-lyon.fr**	M. Jean-Yves BUFFIÈRE INSA de Lyon MATEIS - Bât. Saint-Exupéry 7 Avenue Jean CAPELLE 69 621 Villeurbanne CEDEX Tél : 04.72.43.71.70 Fax : 04.72.43.85.28 **jean-yves.buffiere@insa-lyon.fr**
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** **http://edmega.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction **mega@insa-lyon.fr**	M. Jocelyn BONJOUR INSA de Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69 621 Villeurbanne CEDEX **jocelyn.bonjour@insa-lyon.fr**
ScSo	**ScSo*** **http://ed483.univ-lyon2.fr** Sec. : Véronique GUICHARD INSA : J.Y. TOUSSAINT Tél : 04.78.69.72.76 **veronique.cervantes@univ-lyon2.fr**	M. Christian MONTES Université Lyon 2 86 Rue Pasteur 69 365 Lyon CEDEX 07 **christian.montes@univ-lyon2.fr**

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Abstract

In the last years, credit and debit cards usage has significantly increased. However a non negligible part of the credit card transactions are fraudulent and billions of euros are stolen every year throughout the world. In Belgium alone, the volume of credit card transactions reached 16 billion euros in 2017 with 140 million euros of illegitimate transactions.

Credit card fraud detection present several characteristics that makes it a challenging task. First, the feature set describing a credit card transaction usually ignores detailed sequential information which was proven to be very relevant for the detection of credit card fraudulent transactions. Second, purchase behaviours and fraudster strategies may change over time, making a learnt fraud detection decision function irrelevant if not updated. This phenomenon named dataset shift (change in the distribution $p(x, y)$) may hinder fraud detection systems to obtain good performances. We conducted an exploratory analysis in order to quantify the day by day dataset shift and identified calendar related time periods that show different properties. Third, credit card transactions data suffer from a strong imbalance regarding the class labels which needs to be considered either from the classifier perspective or from the data perspective (less than 1% of the transactions are fraudulent transactions).

Solutions for integrating sequential information in the feature set exist in the literature. The predominant one consists in creating a set of features which are descriptive statistics obtained by aggregating the sequences of transactions of the card-holders (sum of amount, count of transactions, etc..). We used this method as a benchmark feature engineering method for credit card fraud detection. However, this feature engineering strategies raised several research questions. First of all, we assumed that these descriptive statistics cannot fully describe the sequential properties of fraud and genuine patterns and that modelling the sequences of transactions could be beneficial for fraud detection. Moreover the creation of these aggregated features is guided by expert knowledge whereas sequences modelling could be automated thanks to the class labels available for past transactions. Finally, the aggregated features are point estimates that may be comple-

mented by a multi-perspective univariate description of the transaction context (especially from the point of view of the seller).

We proposed a multi-perspective HMM-based automated feature engineering strategy in order to incorporate a broad spectrum of sequential information in the transactions feature sets. In fact, we model the genuine and fraudulent behaviours of the merchants and the card-holders according to two univariate features: the timing and the amount of the transactions. Moreover, the HMM-based features are created in a supervised way and therefore lower the need of expert knowledge for the creation of the fraud detection system. In the end, our multiple perspectives HMM-based approach offers automated feature engineering to model temporal correlations so as to complement and possibly supplement the use of transaction aggregation strategies in order to improve the effectiveness of the classification task.

Experiments conducted on a large real world credit card transaction dataset (46 million transactions from belgium card-holders between March and May 2015) have shown that the proposed HMM-based feature engineering allows for an increase in the detection of fraudulent transactions when combined with the state of the art expert based feature engineering strategy for credit card fraud detection.

To conclude, this work leads to a better understanding of what can be considered contextual knowledge for a credit card fraud detection task and how to include it in the classification task in order to get an increase in fraud detection. The method proposed can be extended to any supervised task with sequential datasets.

Résumé

Au cours des dernières années, l'utilisation des cartes de crédit et de débit a considérablement augmenté. toutefois une partie non négligeable des transactions par carte de crédit sont frauduleuses et des milliards d'euros sont volés chaque année dans le monde entier. Rien qu'en Belgique, le volume de transactions par carte de crédit ont atteint 16 milliards d'euros en 2017, dont 140 millions d'euros. transactions illégitimes.

La détection de fraude par carte de crédit présente plusieurs caractéristiques qui en font une tâche difficile. Tout d'abord, les attributs décrivant une transaction ignorent les informations séquentielles qui se sont avérées très pertinentes pour la détection des fraudes à la carte de crédit. Deuxièmement, les comportements d'achat et les stratégies de fraude peuvent changer au fil du temps, rendant une fonction de décision apprise par un classifieur non pertinente si celui-ci n'est pas mis à jour. Ce phénomène appelé dataset shift (changement dans la distribution de probabilité $p(x, y)$) peut empêcher les systèmes de détection de fraude de conserver une bonne performance. Nous avons effectué une analyse exploratoire afin de quantifier le dataset shift jour par jour et avons identifé des périodes calendaires qui ont des propriétés différentes au sein du jeu de données. Troisièmement, les données sur les transactions par carte de crédit souffrent d'un fort déséquilibre en ce qui concerne les effectifs des classes (moins de 1% des transactions sont frauduleuses). Ce déséquilibre doit être pris en compte, soit par le classifieur, soit au niveau du prétraitement des données.

Des solutions pour intégrer des informations séquentielles au sein des attributs transactionnels existent dans la littérature. La stratégie principale consiste à créer un ensemble d'attributs qui sont des statistiques descriptives obtenues en agrégeant les séquences de transactions des titulaires de carte (somme du montant, nombre de transactions, etc.). Nous avons utilisé cette méthode comme méthode de référence pour la détection des fraudes à la carte de crédit. Cependant, cette stratégie de prétraitement des données a soulevé plusieurs questions de recherche. Tout d'abord, nous avons supposé que ces statistiques descriptives ne pouvaient pas décrire complètement les propriétés séquentielles des motifs

temporels frauduleux et non frauduleux et que la modélisation des séquences de transactions pouvait être bénéfique pour la détection de la fraude. De plus, la création de ces attributs agrégés est guidée par des connaissances expertes, tandis que la modélisation de séquences pourrait être automatisée grâce aux labels de classe disponibles pour les transactions passées. Enfin, ces attributs agrégées sont des estimations ponctuelles pouvant être complétées par une description multi-perspective du contexte de la transaction (en particulier du point de vue du vendeur).

Nous avons proposé une stratégie pour la création d'attributs basés sur des modèles de Markov cachés (HMM) caractérisant la transaction par différents points de vue. Cette stratégie permet d'intégrer un large spectre d'informations séquentielles dans les attributs des transactions. En fait, nous modélisons les comportements authentiques et frauduleux des commerçants et des détenteurs de cartes selon deux caractéristiques univariées: la date et le montant des transactions. De plus, les attributs basées sur les HMM sont créées de manière supervisée, réduisant ainsi le besoin de connaissances expertes pour la création du système de détection de fraude. En fin de compte, notre approche à perspectives multiples basée sur des HMM permet un prétraitement automatisé des données pour modéliser les corrélations temporelles afin de compléter et éventuellement remplacer les stratégies d'agrégation de transactions pour améliorer l'efficacité de la détection.

Des expériences menées sur un vaste ensemble de données de transactions de cartes de crédit issu du monde réel (46 millions de transactions effectuées par des porteurs de carte belges entre mars et mai 2015) ont montré que la stratégie proposée pour le prétraitement des données basé sur les HMM permet de détecter davantage de transactions frauduleuses quand elle est combinée à la stratégie de prétraitement des données de référence basées sur des connaissance expertes pour la détection de fraude à la carte de crédit.

En conclusion, ces travaux permettent de mieux comprendre ce que l'on peut considérer comme une connaissance contextuelle dans le cadre d'une tâche de détection de fraude à la carte de crédit et comment l'inclure dans la tâche de classification afin d'améliorer la détection de fraude. La méthode proposée peut être étendue à toute tâche supervisée comportant des jeux de données séquentiels.

Zusammenfassung

In den letzten Jahren hat die Verwendung von Kreditkarten und Debitkarten beträchtlich zugenommen. Bei einem nicht unerheblichen Teil aller Kreditkartentransaktionen handelt es sich um betrügerische Transaktionen und weltweit werden jedes Jahr Milliardenbeträge gestohlen. Alleine in Belgien erreichte das Gesamtvolumen der Kreditkartentransaktionen im Jahr 2017 16 Milliarden Euro; wovon 140 Millionen auf betrügerische Transaktionen entfallen.

Die Erkennung von Betrug in Kreditkartenzahlungen weist mehrere Besonderheiten auf, wodurch spezielle Herausforderungen entstehen. Zum einen enthält die Merkmalsmenge, die zur Beschreibung einzelner Kreditkartentransaktionen verwendet wird, keine detaillierte Sequenzinformation, die sich jedoch für die Erkennung als höchst relevant herausgestellt hat. Zum anderen können sich sowohl legitimes Kaufverhalten als auch Betrugsstrategien mit der Zeit ändern wodurch eine Entscheidungsfunktion irrelevant werden kann, sofern sie nicht aktualisiert wurde. Dieses Phänomen, bekannt als "data set shift" (eine Änderung der Verteilung $p(x, y)$), beeinträchtigt die Erkennungsleistung von Betrugserkennungsystemen. Wir haben eine explorative Analyse durchgeführt um den täglichen data set shift zu quantifizieren und dabei kalendarische Zeiträume identifiziert, die unterschiedliche Besonderheiten aufweisen. Da Datensätze mit Kreditkartentransaktionen im Hinblick auf die Klassenannotationen ein starkes Ungleichgewicht aufweisen, muss dieses Ungleichgewicht entweder auf der Ebene des Klassifikators oder auf der Ebene der Daten berücksichtigt werden (weniger als 1% der Transaktionen sind von betrügerischer Natur).

Für die Integration von Sequenzinformation in die Merkmalsmenge existieren in der Literatur bereits Lösungen. Die vorherrschende Lösung besteht darin, durch das Aggregieren von Transaktionssequenzen von Karteninhabern, deskriptive Statistiken zu erzeugen (Summe der Geldbeträge, Anzahl an Transaktionen, etc.). Diese Methode zur Merkmalserzugung bildet insofern die Grundlage unser Arbeit als sich daraus mehrere Forschungsfragen ergeben. Zunächst stellen wir fest, dass diese deskriptiven Statistiken die Sequenzeigenschaften in betrügerischen oder legitimen Mustern nicht vollständig abbilden können und dass

eine Modellierung von Transaktionssequenzen vorteilhaft für die Betrugserkennung wäre. Außerdem erfordert die Erzeugung solcher aggregierter Merkmale Expertenwissen wohingegen Sequenzmodellierung dank der vorhandenen Klassenannotationen automatisiert werden könnte. Aggregierte Merkmale sind Punktschätzungen, die mit univariaten Beschreibungen des Transaktionskontextes aus unterschiedlichen Perspektiven ergänzt werden könnten (insbesondere aus der Perspektive des Händlers).

Wir schlagen eine multi-perspektivische HMM-basierte und automatisierte Strategie zur Merkmalserzeugung vor um ein breites Spektrum an Sequenzinformation in die Merkmalsmenge zu integrieren. Wir modellieren die legitimen und die betrügerischen Verhaltensweisen von Händlern und Karteninhabern auf Grundlage zweier univariater Merkmale: Die Zeitstempel und die Geldbeträge von Transaktionen. Die HMM-basierten Merkmale werden zudem unter dem Paradigma des überwachten Lernens erzeugt wodurch der Bedarf an Expertenwissen bei der Entwicklung eines Betrugserkennungssystems sinkt. Schlussendlich erzeugt unser HMM-basierter Ansatz zeitliche Korrelationsmerkmale automatisiert wodurch Aggregationsstrategien ergänzt oder möglicherweise ersetzt werden können und die Effektivität der Klassifikation verbessert werden kann.

Experimente auf einem großen und realistischen Datensatz (46 Millionen Transaktionen aufgezeichnet zwischen März und May 2015 von Karteninhabern aus Belgien) haben gezeigt, dass die HMM-basierte Merkmalserzeugung die Erkennung betrügerischer Transaktionen verbessert sofern die HMM Merkmale mit state-of-the-art Expertenmerkmalen kombiniert werden.

Diese Arbeit führt zu einem besseren Verständnis davon was als kontextuelles Wissen in der Erkennung von Kreditkartenbetrug betrachtet werden kann und wie solches Wissen in die Klassifikation integriert werden kann um die Erkennung zu verbessern. Unsere Methode kann auf andere annotierte Sequenzdaten übertragen werden.

Acknowledgements

First, I would like to express my sincere gratitude to my supervisors: Dr. Pierre-Edouard Portier, Dr. Léa Laporte, Prof. Michael Granitzer and Prof. Sylvie Calabretto for the continuous support throughout my PhD study, for their patience, motivation, and immense knowledge. Their guidance helped me in all the time of research and writing of this thesis. Having 4 supervisors was challenging: you have to consider and test all the various directions proposed, but it was also very valuable in several ways: you learn from different persons and discover different working and researching patterns. I could not have imagined having better supervisors and mentors for my PhD study.

Besides my advisors, I would like to thank Prof. Harald Kosch, Prof. Lionel Brunie, Dr. Nadia Bennani, Dr. Gabriele Gianini and all the people that make IRIXYS (formerly MDPS) live and be an insightful and friendly community. Without them, this french-german PhD wouldn't exist, neither the friendships that were born in this context.

Speaking of friendships, I would like to thank all the friends that I met during the thesis and especially my 4 year office mate, project mate and in fine PhD mate, Johannes Jurgovsky, with whom we could share difficulties, joys, discoveries and a lot of beers (a good meeting point between Bavaria and Brittany). I hope and am confident that some of these friendships will last whatever the years or the distance.

My sincere thanks also go to the people from Worldline and especially Dr. Olivier Caelen, Dr. Liyun He-Guelton and Dr. Frederic Oblé, who provided me the opportunity to work within a company, with real data and stakes.

Lastly, I would like to thank my family for all their love and encouragement. For my parents who raised me with a love of science and believed in me whatever the challenge I chose. For the presence of my brother Marin and my sister Katell who also challenged themselves for their studies and first work experiences. And most of all for my loving, supportive, encouraging, and patient Lisa. Thank you for your faithful support during all the stages of the PhD, thank you for the sacrifices you did in order to follow me in Passau for a year. These years wouldn't have been the same without you and I am very happy to have met you. Thank you.

Scientific environment

The research reported in this dissertation has been done at the Chair of Data Science of the University of Passau (Germany) and at the research group for Distributed Systems, Information Retrieval and Mobility (DRIM of the Laboratory of Computer Science for Image and Information Systems (LIRIS) at INSA Lyon (France).

This research has received support from *Worldline* in the form of an industrial research project including *Worldline*, the University of Passau, INSA Lyon and the University of Milano.

As a cotutelle between INSA Lyon and University of Passau, this research has also received financial support from the Franco-German university

Science sans conscience
n'est que ruine de l'âme.

Rabelais

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Credit card fraud in Europe

In 2011, 700 millions payment cards have been issued in the EU. The volume of non-cash transaction for that year exceeded 3000 billion euros. The implementation of EMV[1] (chip-embedded cards) for face-to-face transactions and strong identification of customers via 3D secure[2] for e-commerce transactions increased significantly the security of european payments.

However, even after the implementation of EMV and 3D-SECURE security, a non negligible number of credit card transactions remains illegitimate: the amount of european credit card fraud reaches 1.5 billion euros yearly. In order to complement the decrease in credit card fraud obtained with the inclusion of authentication methods, experts agree that data driven fraud detection systems are the future of credit card fraud detection [Ali et al., 2019].

According to [Europol, 2012][3], international organised crime groups dominate the criminal market of credit card fraud and affect non-cash payments on a global level. Doing fraudulent card payment is not risky and provide high profit to these organised crime groups. These incomes are afterwards invested in order to develop further fraudulent strategies, to finance other criminal activities or to start legal businesses (money laundering).

Experts reported that the international and highly-organized nature of criminal networks creates the need for international police cooperation. However, the

[1] EMV (Europay, MasterCard, Visa) : a standard for payment cards based on chip technology.

[2] 3D secure: double identification of the card-holder via a PIN sent by SMS

[3] EUROPOL: European agency for criminality repression and collaboration between police forces

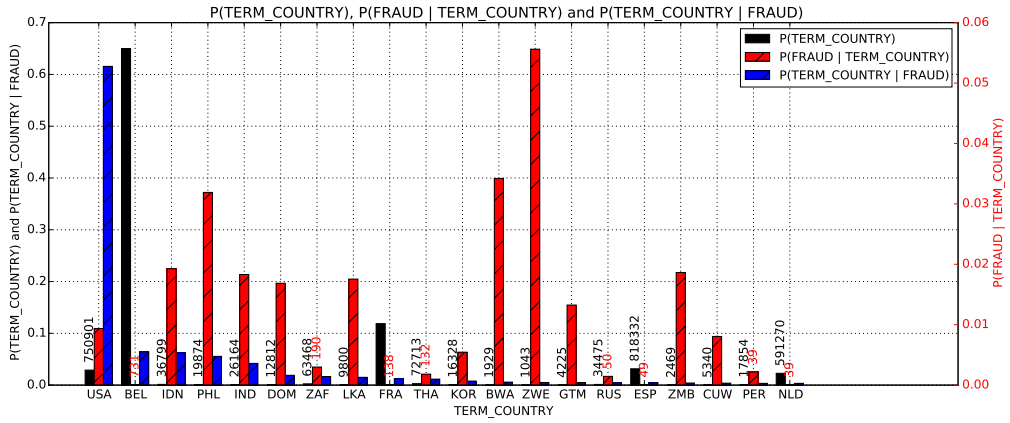


Figure 1.1: Repartition of different countries among the belgian face-to-face transactions between march and may 2015 (*(Blue bars refers to the repartition of the different countries among fraudulent transactions (scale on the left side). USA is by far the most represented country among fraudulent countries)*

legal constraints and limitations for law enforcement authorities in order to fight credit card fraud makes it a very attractive business for organized crime groups. EUROPOL [Europol, 2012] indicated that the same criminals are still active after many years and return after a few months to the business. Despite this, if a global operation involving the cooperation of the police forces of several states is conducted, it has a big impact on the security of credit card transactions in the EU. For example, in June 2011, a operation named *'Night Clone 11'* led to the arrestation of 70 suspects in EU and overseas. This operation initiated by the Italian Police with the cooperation of Bulgaria, US Secret Services and other EU Member States and coordinated in the operational center of EUROPOL caused a break in illegal activities for many other organized crime groups for several months.

Figure 1.1 shows the repartition of the feature TERM-COUNTRY (refering to the country of origin of the merchant) for belgian face-to-face transactions in 2015. The blue bars refer to the repartition of this feature categories among the fraudulent transactions only whereas the black bars refer to the repartition of this feature categories among all transactions. Their scale is on the left. We can observe on figure 1.1 that surprisingly, the majority (60%) of fraudulent face-to-face transactions affecting the European Union cardholders takes place overseas. This is mostly due to the fact that EMV standards are not adopted worldwide as a global solution against the counterfeiting of payment cards. Indeed, since the introduction of the EMV technology in 2008, the losses caused by domestic fraudulent transactions drastically decreased in the EU. However at the same time

the level of illegal transactions overseas has seen a big increase. Consequently, since 2011, almost all fraudulent transactions happen overseas. The countries that are the most important in terms of the number of fraudulent transactions are:

- United States,
- Dominican Republic,
- Colombia,
- Russian Federation,
- Brazil,
- Mexico.

In the end, data-driven methods may find and leverage patterns in credit card fraudster behaviours like the aforementioned merchant country pattern. These data-driven methods would complement existing authentication solutions like EMV authentication for face-to-face transactions or 3D-SECURE authentication for e-commerce transactions.

1.2 Credit card fraud types

Credit card frauds are categorized in various ways in the literature. [Delamaire et al., 2009] proposes to differenciate the fraud with respect to the fraudsters strategies. They split them into application frauds and behavioral frauds. In application frauds the fraudsters apply for a credit card with a false ID whereas in behavioral frauds the fraudsters find a way to obtain the cardholder's credential in order to use a pre-existing credit card. [Ghosh and Reilly, 1994] split the fraudulent transactions into six categories with respect to the fraudulent process: frauds from lost or stolen cards, frauds from counterfeit cards, online frauds, bankruptcy frauds, merchant frauds and frauds from cards that got stolen during the expedition process. [Patidar et al., 2011] split the fraudulent transactions in three categories: card related frauds, merchant related frauds and Internet frauds. [Laleh and Azgomi, 2009] go further in this direction and propose to only split fraud into face-to-face (card present) fraud and e-commerce (card not present) fraud. Their argument for this strict and simple classification is that the overlap between categories may weakens fraud detection approaches.

As described by credit card fraud detection experts, in the industry the fraudulent scenarios are split in 5 different types:

- Lost/stolen cards (≈1% of fraudulent transactions): Mostly against elderly cardholders, the fraudster get the PIN code by shoulder reading and steal the card afterwards. In this case the fraudster is the thief, the credit card doesn't pass through a reselling network from organized crime.

- Non received cards (<1% of fraudulent transactions): Credit card stolen during production or postal delivery. In order to avoid these type of fraud, the banks can ask the customer to retrieve his card at the bank agency, or to call them in order to activate the card.

- ID Theft (marginal): Card obtained using false or stolen ID documents.

- Counterfeint cards (<10% of fraudulent transactions): The card is copied during a genuine card usage or during database hacking and reproduced afterwards on fake plastic by international organized crime groups. The fraudster get and reproduce the data of the magnetic stripe of the cards. This type of fraud was predominant in the past but partly solved by the EMV technology: magnetic stripe only terminals aren't used anymore in EU but remain in Asia and America. It is worth noting that contactless payment isn't very interesting for fraudster since only low amount payments are allowed.

- Card not present frauds (>90% of the fraudulent transactions): Most of the credit card frauds happen on e-commerce transactions. The credentials (card number, expiry date and CVC) are usually retrieved during a database hacking organized by international crime groups and are afterwards sold on the dark web. British airways, Mariot Hotels and Ticket Master were for example victims of data breaches in 2018. The price of the credentials depends on the facility of doing fraud with them (the first digits of the card numbers identify the bank and therefore the blocking policy). Most merchants (90%) use the 3D SECURE technology which protect the cardholder with a double identification however some major merchant website such as Ebay or Amazon don't protect their users with 3D SECURE. An other problem that hinder fighting against card not present fraud is that companies don't report that there was an attack that caused data breaches because of the bad advertising it would cause.

The means for the fraudster to acquire card data are various. Firstly, for card non present fraud, illegal intrusion in the credit card database are a way to gain card data as stated above. Phising can also be used in order to get data for e-commerce payment. It consists in tricking the card-holder with an email for example in order to redirect him to a false bank website and get his identifiers. Furthermore, there are tools that have been developed in order to

generate card numbers (see figure 1.2 for an example of these tools). These numbers are then used for "credit master attack", which consists in brute force attacking the merchant website with lots of possible cards (the expiry date and CVC can not be inferred and need to be brute forced too).

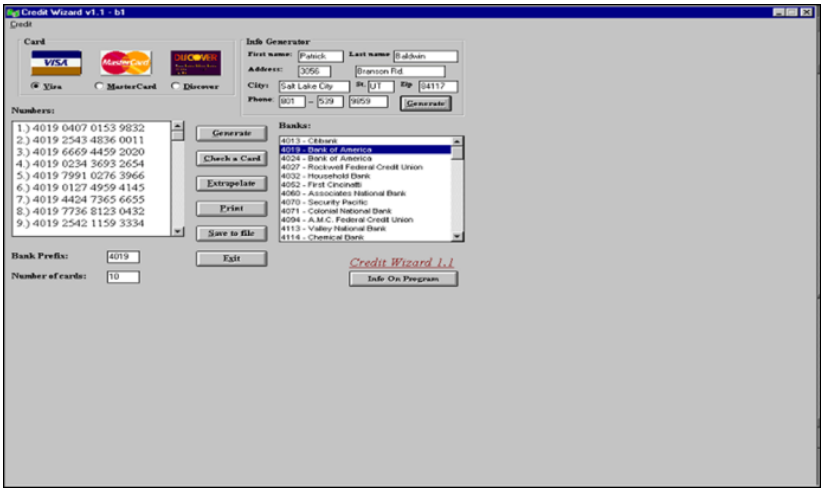


Figure 1.2: Card number generator

For card present frauds, fraudsters may use a sleight of hand in order to swap credit cards or a device to block the credit card in the ATM after having read the PIN by "shoulder surfing". An other way can be skimming or shimming credit card by recording magnetic stripe data for later reproduction or more recently attach a false card reader device on top of the terminal (see figure 1.3).

Figure 1.3: Cardblocker PIN reader terminal topper

The different fraudulent transactions types aforementioned could hinder the ability of a data driven system to model the various behaviours leading to a fraud.

In order to partially solve this issue, e-commerce and face-to-face transactions are split in this work as adviced by [Laleh and Azgomi, 2009] since the genuine and fraudulent behaviours show very different properties for e-commerce and face-to-face.

1.3 Fraud detection in practice

This PhD work was done in collaboration with Worldline, a french multinational information technology service and consulting company specialized in high-tech transactional services, cloud, big data and cybersecurity services. In this section we will describe how credit card fraud detection is done in practice as it was described to us during an interview with an expert.

1.3.1 Real time and near real time fraud detection systems

In practice, credit card fraud detection is a process that happens in two steps: the blocking time that aims to detect fraudulent transactions and the checking time that aims to detect fraudulent cards (see figure 1.4). These two steps are very different in term of time constraint: the blocking time must be short since the card-holder is waiting for his transaction to be accepted whereas the checking time can length more.

The blocking time corresponds to the real time fraud detection. It aims to authorize or not the transactions. This process must be fast and very precise. Indeed, it must block only transactions that are very likely to be fraudulent, otherwise it would hinder the cardholder buying routine. Most of the transactions blocked by the real time fraud detection systems are fraudulent transactions, on the other hand, a lot of fraudulent transactions are not detected. The priority of the real time fraud detection system is precision, not recall (see section 2.2.1 for metrics description). For rapidity purposes, it is made of expert rules based on simple and short term aggregations of the past transactions of the card holder.

The checking time corresponds to the near real time fraud detection. In the hours following the authorized transaction, the near real time fraud detection system combines expert rules based on long term aggregations and machine learning algorithms to fire alerts on credit cards that present suspicious transactions. Afterwards, alerts are usually checked by expert investigators, or automatic SMS are sent to the cardholder to invite them to verify their recent transactions and block the card if needed. The cardholder is sometimes called in order to label all the recent transactions in order to know when the fraud started.

Since it is impossible to block every credit card with a suspicion of fraud and

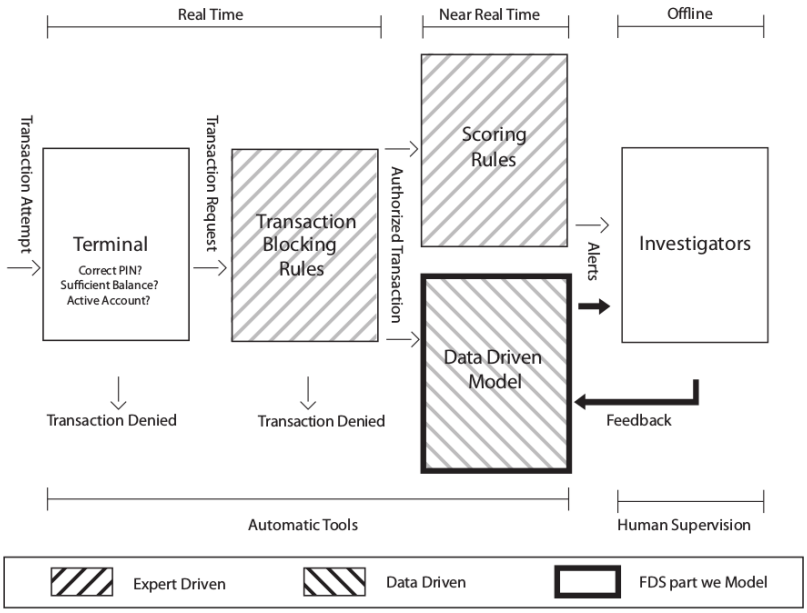


Figure 1.4: Real time and near real time fraud detection systems [Pozzolo, 2015]

very costly for experts to investigate every alert raised by fraud detection system. [Baesens et al., 2015] highlighted the fact that investigating alerts raised by fraud detection systems is costly. They showed that a tradeoff strategy can be used between the probability of the alert to be a fraud (How sure the classifier is that the alert is a fraud) and the amount of money that can be saved by preventing this fraud (see figure 1.5)). Depending on the uncertainty of the fraud detection system that a transaction is fraudulent and the amount of the transaction, the experts can automatically decide to investigate or not.

1.3.2 Rules based detection and rules management

After alerts are processed by the fraud case management team, the feedback of the investigators is used in order to update the rules and the systems.

There are two different types of rules:

- filter rules: Automatically decline transactions based on the transaction characteristics. This helps to raise alerts for known fraudulent patterns. For example: *reject transactions with amount superior to 300€ in the country code 392 (japan) with merchant name containing "AEON", with mercant category different from 6011 and with card number different from XXX and YYY (2 cards excluded from the rule).*

- history based rules: An aggregation function is calculated and the rule decision is based on a threshold on the value of the aggregation.

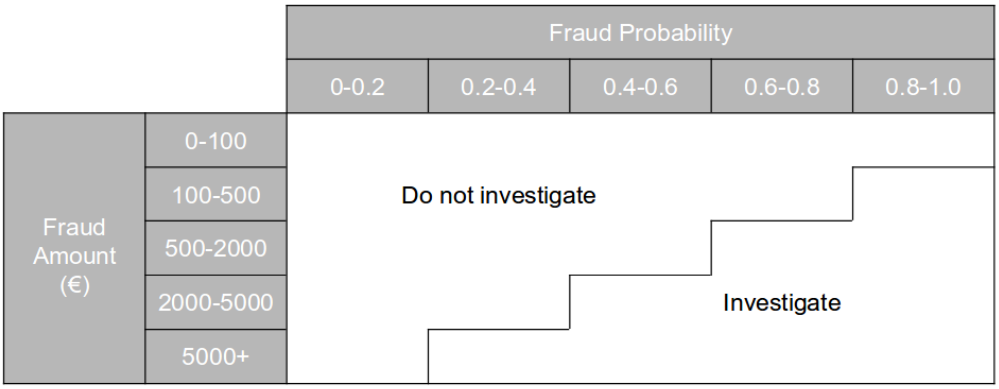


Figure 1.5: Tradeoff between fraud probability and transaction amount (from [Baesens et al., 2015])

At the end of the month, several reports of the efficiency of the rules are produced. The main report aims to tell, for each rule: the number of false positives, true positives and the monetary savings from the rule. Among the rules contained in the fraud detection systems, some may become out of date and up to date again afterwards. New expert rules are created by fraud investigators and calling agents: They monitor a lot of different fraudulent transactions each days and recognize common patterns between fraud scenarios. These similarities are included in the expert rules in order to raise alerts.

When a fraudulent transaction is detected and investigated, the previous transactions of the cardholder are carefully labelled with respect to the time elapsed between until the detected fraudulent transaction. But there is a grey zone with potentially a mix of genuine and fraudulent transactions.:

Conclusion In the end, double identification through 3D-SECURE for e-commerce transactions or EMV PIN technology for face-to-face transactions strongly increased the security of credit card transactions. Moreover, the combination of real time expert rules (automatic transactions blocks), near real time expert rules (manual cards blocks) and data driven alerts allow for an appreciated prevention and detection of fraudulent transactions.

However, fraudulent transactions still represent a significant loss for customers and bank companies that have to refund them. Moreover, fraudulent strategies are constantly changing in order to get around fraud detection systems. Besides, the double authentication processes aforementioned are not implemented in several major countries and web vendors, creating the need for improved fraud detection systems.

According to industrial partners, advanced data analytics and data driven methods are very promising for fraud detection and would lead to savings from the company along with a more trustful experience for their customers.

1.4 Publications and outline

1.4.1 Publications

Data driven credit card fraud detection presents several characteristics that makes it a difficult task. They will be described more in depth in section 2.3. Briefly:

- Fraud detection's main characteristic is that there is a strong imbalance between the classes: the fraudulent events are less represented than the genuine ones.
- Fraudsters may adapt their strategies over time. Besides buying behaviour may change seasonally. Therefore, a credit card fraud detection system may become out of date if no action is taken.
- Even if credit card transactions may present some conditional dependences, credit card transactions usually ignore detailed sequential information. Feature engineering is a crucial issue for increasing fraud detection system's performances.

Throughout this thesis, we attempt to answer these research questions. The answers we proposed were opportunities for scientific publications.

The main contribution of the thesis, the multiple perspective HMM-based feature engineering framework, introduced in section 2.3.2 and described in chapters 5 and 6 was published in:

- Multiple perspectives HMM-based feature engineering for credit card fraud detection, *34th ACM/SIGAPP Symposium on Applied Computing (SAC2019)*. [Lucas et al., 2019a]
- Towards automated feature engineering for credit card fraud detection using multi-perspective HMMs, *Future Generations Computer Systems Special Issue on: Data Exploration in the Web 3.0 Age.*[Lucas et al., 2019b]
- Additionaly, the source code of the proposed framework can be found at `https://gitlab.com/Yvan_Lucas/hmm-ccfd` .

Additionally, a study of the calendar based covariate shift observed in the credit card transaction dataset was performed. It is described in the section 4.2 of this work and has been presented in:

- Dataset shift quantification for credit card fraud detection, *IEEE Artificial intelligence and knowledge engineering (AIKE 2019).* [Lucas et al., 2019c]

1.4.2 Outline

This work is articulated as following: after having introduced the stakes of credit card fraud detection and the properties of this task in chapter 1, we will describe more formally the problem of credit card fraud detection and highlight the research questions it raises in chapter 2. Afterwards, we will present in chapter 3 the state of the art approaches for handling the special difficulties of credit card fraud detection (namely: dataset shift, sequence modelling, class imbalance and feature engineering). In chapter 4, we will highlight the particularities of the credit card transactions dataset used throughout this work and conduct an exploratory analysis in order to quantify the dataset shift in it. In chapter 5 we will present the multiple perspective HMM-based feature engineering strategy which is the backbone of this work. The increase in fraud detection obtained with the proposed HMM-based features will be shown in chapter 6.

Chapter 2

Problem statement & research questions

We introduced in the previous chapter the real world stakes that motivated credit card fraud detection. Moreover, some informations about fraud taxonomy and how it is leveraged in practice have been presented. More precisely, the credit card fraud detection happens in two times: at the time of the transaction there is a real-time fraud detection system that detects obvious frauds, and afterwards, histories of transactions are evaluated in order to raise alerts on suspicious credit card centered buying behaviours.

In this chapter, we start with a formalization of the problem of credit card fraud detection as a machine learning task along with an introduction of the ensemble models used throughout this thesis: the random forest classifier and the boosting trees classifier. We also briefly illustrate the strengths of ensemble-based methods.

Then we introduce two ways of evaluating machine learning classifiers: the confusion-matrix based evaluation and the parametric evaluation, and argue the choice of parametric evaluation for credit card fraud detection.

We finish this chapter by describing the questions that make credit card fraud detection a challenging research task.

2.1 Machine learning formalization for credit card fraud detection

2.1.1 Machine learning notations & Fraud detection particularities

Fraud detection formalization

Throughout this work, we will refer to x as the set of features describing an example (the set of attributes describing a transaction for our particular problem). Each feature x_k of the feature set can be categorical, ordinal or numerical.

A categorical feature takes values in a discrete non-ordered ensemble of categories $\{c_1, .., c_n\}$. For the credit card fraud detection application, a categorical feature can be for example: the country of the merchant involved in the transaction.

An ordinal feature takes values in a discrete ordered ensemble of categories $\{c_1, ..., c_n\}$ with $c_i < c_j$ if $i < j$ $(i, j \in \{1, ..., n\}^2)$. For the credit card fraud detection application, an ordinal feature can be for example: the credit limit of the card-holder that made the transaction: it has a fixed number of categories corresponding to the usual values of credit limit but these categories can be ordered.

A numerical feature takes continuous values. The application domain of a numerical feature can be bordered or non bordered: its values can be limited to an interval or not. For the credit card fraud detection application, a numerical feature can be for example: the amount of the considered transaction.

y will refer to the target variable: a feature for which we aim to know the value conditioned on the set of features x. y can be categorical or continuous. For our particular problem the target variable will most often be the ground truth for the fraudulent nature of the transaction: Is it a fraudulent transaction or not?

Machine learning algorithms aim to model the dependencies between the feature set x and the target variable y. More precisely, discriminant algorithms aim to model the conditional distribution of probability $p(y|x)$ whereas generative algorithms aim to model the joint distribution of probability $p(x, y)$. In the case of a categorical target variable, the discriminant model is called a classifier, whereas in the case of a continuous target variable, the discriminant model is called a regressor.

Machine learning consists in two main phases:

- During the training phase, the model is trained in order to be able to make a prediction for a particular task. The particular task is defined by a set of training examples representing the function that the model aims to approximate. These training examples are usually preprocessed before being given to the model in order to make sure that the model fits his decision function using appropriate information from the data. At the end of the training phase, the model parameters are adjusted in order to reflect the most the relationship between the set of features x and the target variable y observed in the training set of examples. An additional step called validation step can be performed in order to adjust the model hyperparameters[1]

- The testing phase aims to leverage the knowledge acquired by the model during the training phase. The trained model is evaluated on a set of examples **distinct** from the set of examples used for training.

Dataset

Throughout this work, we study a confidential dataset containing all the credit card transactions issued by belgian credit cards between 01.03.2015 and 31.05.2015.

As mentioned before, credit card transactions can be separated in two very different clusters:

face-to-face The face-to-face transactions are transactions for which the card-holder is located at the shop. Thanks to the EMV technology aforementioned, we observe in the dataset that the authentication of the card-holder is made with a PIN code 92% of the time. However in 7% of the face-to-face transactions, the authentication is made with a signature. The proportion of fraudulent transactions is 20 times higher with signature authentication than with PIN authentication. Besides, we observe that 82% of the fraudulent face-to-face transactions are comprised within the 8% non EMV transactions. Furthermore, face-to-face transactions are restricted in time: very few face-to-face transactions are issued between 12 p.m. and 7 a.m.

e-commerce The e-commerce transactions are remote transactions for which the card-holder is not necessarily located at the shop. We observe that 22% of the belgian e-commerce transactions during spring 2015 have triggered a 3D-SECURE authentication. 3D-SECURE authentication consists in asserting the identity of the card-holder by double checking it, usually via a code sent by SMS. There are 32 times more e-commerce fraudulent transactions without 3D-SECURE authentication than with 3D-SECURE au-

[1]Hyperparameters are fixed parameters usually ruling model architecture. Contrary to the parameters, the hyperparameters are not modified during the training of the model.

Table 2.1: Class representations in the data-set

	Fraud	Non-Fraud	Total
Face-to-Face	10.815	25.824.400	25.835.215
E-commerce	73.687	21.524.624	21.598.311

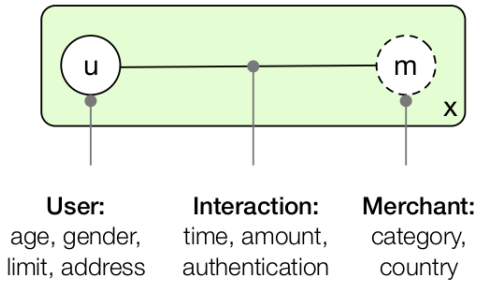


Figure 2.1: Bipartite graph of the transactions.

thentication. Furthermore, a significantly bigger proportion of e-commerce transactions is issued at night compared to face-to-face transactions: around 10% of the e-commerce transactions are issued between 12 p.m. and 7 a.m. whereas only 3% of face-to-face transactions are done within the same period of time. The buying schedule is more spread for e-commerce transactions

There is a significant difference in fraud ratio between e-commerce and face-to-face transactions. We observe in the Belgian credit card transactions dataset that there are 4 fraudulent transactions for 10.000 face-to-face transactions and 3 fraudulent transactions for 1000 e-commerce transactions (see table 2.1). The class imbalance is 17 times stronger for the face-to-face transactions than for the e-commerce transactions

E-commerce and face-to-face transactions are very different paradigms and therefore are studied separately in this work, moreover classifier efficiencies are measured on the two types of transactions.

Terminal/Card-holder sequence

Credit card transactions are an interaction between two nodes of different types: a card-holder node and a merchant node. Transactions can be seen as edges linking merchant and card-holder in a bipartite graph (see figure 2.1).

Moreover, the time of the transaction is recorded. The transactions can be ordered with respect to the time scale. Therefore, transactions of a given card-

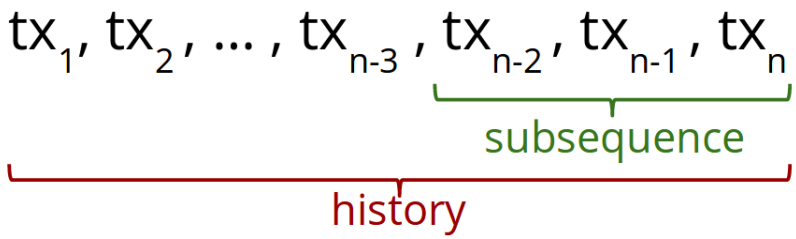


Figure 2.2: history and subsequence of transactions

holder or merchant can be gathered and assembled together in sequences.

Since we stated earlier that e-commerce and face-to-face transactions are very different paradigms, we made the hypothesis that e-commerce and face-to-face transactions are not related within a sequence: each card-holder have a face-to-face sequence of transactions and an e-commerce sequence of transactions.

In this work, we name card-holder (resp. merchant) **history** the sequence containing all the previous transactions of a card-holder (resp. merchant). The card-holder history $H_{ch_i}(t)$ comprises all the transactions of the card-holder happening before the current transaction (e.g. the transaction currently waiting for decision in the fraud detection system).

Furthermore, we name card-holder (resp. merchant) **subsequence** a sequence containing a subset of transactions happening before the actual transaction (see figure 2.2. The transactions in the subset of transactions are happening just before the current transaction. The number of transaction in the subsequence is either chosen arbitrarly or depends on the number of transactions that have happened in a given amount of time.

The sequential view can be leveraged for a better knowledge of the context of a transaction: the properties of this transaction can be related to the previous transactions of the sequence.

2.1.2 Bias-Variance tradeoff and ensemble based methods

It is mentioned in the previous sections that the credit card fraud detection classifier aims to estimate the posterior probability $p(y|x)$ (with y the target variable and x the attributes of the transaction).

Since the posterior probability is continuous, this probability estimation problem can be seen as a regression problem that aims to find a function $\hat{f}$ that approximates the underlying function f such as $y = f(x) + \epsilon$ (with ϵ a centered

gaussian noise of variance σ^2). This formulation will ease the demonstration. For evaluation purpose, we want this function $\hat{f}$ to minimize the mean squared error among the test examples:

$$\begin{aligned} MSE &= \mathbb{E}[(y - \hat{f}(x))^2] \\ &= (\mathbb{E}[\hat{f}(x)] - f(x))^2 + \mathbb{E}[\hat{f}(x)^2] - \mathbb{E}[\hat{f}(x)]^2 \\ &= (Bias[\hat{f}(x)])^2 + Var[\hat{f}(x)] + \sigma^2 \end{aligned}$$

- The bias is the difference between the model prediction $\hat{f}(x)$ and the correct output of the underlying function $f(x)$ which we are trying to predict. Simple models usually have high bias since they don't fit a lot to the training data due to less parameters. They usually have similarly high error on training and test data.

- The variance is the variability of the model prediction $\hat{f}(x)$ for different training sets. A model with high variance pays a lot of attention to the training data and will change a lot his prediction when trained on a different training set. Therefore, models with high variance usually perform well on training data but have high error rates on testing data.

- The term σ^2 is the irreducible error coming from the gaussian noise of the world ϵ.

Typically, complex models have high variance and a low bias while simple models have low variance and a high bias. This is because complex models can approximate more the target function (low bias), but are highly affected by the variability of the training set (leading to high variance). The opposite occurs with simple models.

The bias variance tradeoff is a key machine learning problem. Users want their models to be able to identify particular patterns in their data without learning unnecessary and irrelevant relationship between the attributes x and the labels y of the training data. A model that doesn't identify particular patterns in the training data is said to be underfitting. On the other hand, a model that learns unnecessary and irrelevant patterns of the training data is said to be overfitting (see figure 2.3).

[Bauer and Kohavi, 1999] have found that ensembles of model perform often better than a single model. They stated that combining models (and their variances) can lead to a smaller variance overall than single models. Bagging [Breiman, 1996] is the most famous ensemble based methods.

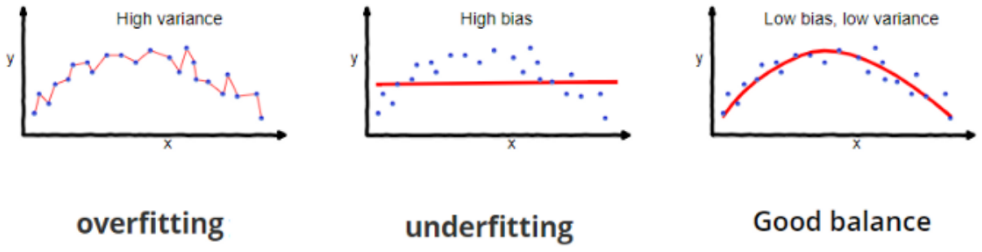


Figure 2.3: overfitting vs underfitting.

Bagging consists in building several estimators $\hat{f}$ on different boostrap samples of the training data. These estimators predictions are then combined. [Breiman, 1996] showed that combining unbiased classifiers with bagging can lead to a nearly optimal one thanks to the reduced variance term.

2.1.3 Random forest and boosting trees

Random forest and boosting trees are two ensemble based approaches that are based on decision tree classifiers. They showed good results for credit card fraud detection and are used at several occasions throughout this work.

Decision tree classifier

Decision trees are a flowchart structure that consists in a succession of internal nodes each containing a rule and branches representing the outcome of the rule (see figure 2.4).

As shown in figure 2.4, one of the asset of decision trees is that it is a model that is very easy to interpret: it is possible to have an understanding of the classifier decision by observing the splitting decisions and their outcomes.

The main metric used in order to find the best rule at a given node is Gini impurity. Gini impurity for a set of element with N classes is calculated as follow (with p_i the ratio of element of class i in the set of elements):

$$Gini = \sum i = 1^N p_i * (1 - p_i)$$

It measures how often a random element would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the set of element. The best split at a given node is obtained by finding the rule that gives the highest Gini impurity decrease through this node.

There existed several iterations of trees implementations:

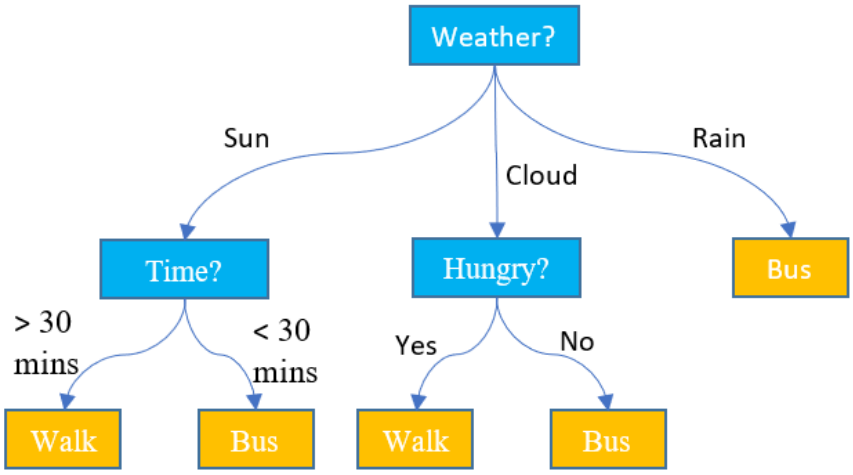


Figure 2.4: Toy example of decision tree [Hoare, 2019]

ID3 ID3 (Iterative Dichotomiser 3) creates a tree by finding for each node the categorical feature that will provide the largest information gain for the categorical target variable. Trees are grown to their maximum size and then a pruning step is usually applied to improve the ability of the tree to generalise to unseen data.

C4.5 C4.5 is the successor to ID3. The categorical feature restriction of ID3 is solved by dynamically defining a discrete variable based on numerical variables that partitions the continuous attribute value into a discrete set of intervals. C4.5 converts the trees with the ID3 algorithm into sets of rules. The accuracy of each rule is then evaluated to determine the order in which they should be applied: the more accurate rules are applied first (at the beginning of the tree). Pruning is done by removing a rule if the accuracy of the parent rule is higher without the child rule.

CART CART (Classification and Regression Trees) is very similar to C4.5, the main difference with C4.5 is that it supports numerical target variables (regression).

Decision trees main issue is that they are very subject to overfitting [Kingsford and Salzberg, 2008]. This is due to the data being repeatedly split into smaller subsets. Overfitting may be mitigated by pruning and early-stopping.

Random forest classifier

Random forest [Breiman, 2001] is a well established algorithm based on bagging. It consists in an ensemble of CART decision trees trained on different views of the training data.

The particularity of random forest is that each tree is trained using a random subset of features. Therefore, each tree reflects a particular decision process based on a subsample of the available information. Moreover, the trees are trained by default on different slices of the dataset thanks to bootstrap sampling[2]. In the end, the trees are very different and the ensemble based model doesn't overfit.

The main parameters of the random forest classifier are:

- The number of features randomly chosen in order to build each tree. An often recommended default value is the square-root of the total number of features.
- The depth of the trees. Contrary to a decision tree classifier, a tree of maximum depth doesn't lead to overfitting since the predictions of very different trees are aggregated together thanks to the ensemble based architecture. However, tree pruning diminish the computation time needed for training and testing.
- The number of trees built. Usually, an higher number of trees leads to a better prediction by the random forest model.

The prediction of a random forest classifier is obtained by averaging all the individual trees predictions for regression tasks or by majority voting for classification tasks.

Random forest classifiers present an interesting property named 'feature importance' that enables users to understand qualitatively the decision of the classifier. This property is leveraged in section 4.1.2.

The robustness to variance along with the simplicity of interpretation of random forest decision makes it a model of predilection for various applications including credit card fraud detection. Moreover, the fact that the trees are built independently from each other allows parallelization of random forest training and testing steps.

[2]Bootstrapping is a type of resampling where large numbers of smaller samples of the same size are repeatedly drawn, with replacement, from a single original sample.

Adaboost and Gradient boosting methods

Boosting is an other ensemble based method that consists in iteratively compensate the misclassification of the ensemble classifier by adding new classifiers (decision tree throughout this work) to it.

There are two approaches for compensating the misclassification of the algorithm: Adaboost and gradient boosting.

- Adaboost changes the samples distribution at each iteration. It increases the weights of the misclassified examples and decreases the weights of the correctly predicted examples. Therefore, the weak learner trained at a given iteration focuses especially on the difficult examples at this iteration. After being trained, the weak learner is added to the ensemble classifier according to his performance: its weight in the decision process depends on how good it is by itself.

- Gradient boosting doesn't change the samples distribution. Instead, each new weak learner aims to learn the remaining error of the ensemble classifier. At each iteration i, the target variable of the weak learner $y\star$ is equal to $y - pred_{i-1}$ with y the ground truth label of each example and $pred_{i-1}$ the value of y guessed by the ensemble classifier at the iteration $i-1$. Each weak learner aims to compensate the gap between the prediction of the ensemble classifier and the ground truth y.

Boosting approaches are prone to overfitting since they aim to iteratively reduce the error of the model on the training set to 0[Quinlan, 1996]. Therefore, tree pruning and early stopping criterion are needed for preventing overfitting.

2.2 Evaluation of the performance of machine learning classifiers

2.2.1 Confusion matrix based evaluation

Traditional ways of evaluating machine learning classifiers use the confusion matrix describing the difference between the ground truth of the dataset and the model prediction (see table 2.2). A perfect model would have all the positive example predicted positive and all the negative example predicted negative: FN and FP would be equal to 0.

Using the confusion matrix, one can compute a variety of metrics in order to compare classifiers performances:

Table 2.2: Confusion matrix

	predicted positive	predicted negative
actual positive	True Positive (TP)	False Negative (FN)
actual negative	False Positive (FP)	True Negative (TN)

Precision Proportion of well classified examples among examples of the positive class:

$$prec = \frac{TP}{TP + FP}$$

Recall Also called true negative rate, proportion of examples of the positive class well-classified among all positive examples:

$$rec = \frac{TP}{TP + FN}$$

False Positive Rate Quantity used for computing the ROC curve (see section 2.2.2):

$$fpr = \frac{FP}{FP + TN}$$

True Positive Rate Quantity used for computing the ROC curve:

$$tpr = \frac{TP}{TP + FN}$$

Accuracy Proportion of well classified examples among all the testing examples (similar to precision but for every class):

$$acc = \frac{TP + TN}{TP + TN + FP + FN}$$

F1 score Precision and recall have opposite behavior: forcing a good precision is detrimental for the recall and vice versa. F1 score is the harmonic mean between precision and recall and therefore gives equal importance to precision and recall. For this reason, this metric which takes values between 0 and 1 is often considered a good by default metric for unbalanced classification tasks:

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} = 2 * \frac{2TP}{2TP + FP + FN}$$

MCC Matthews correlation coefficient is another score that is robust to imbalance [Power, 2011]. It is related to the chi-square statistic and takes values between -1 and 1. For binary classification:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Confusion matrix based metrics are a good starting point for evaluating classifiers performances. However these metrics are point estimators and some users may want to have a fine grained knowledge of how the classifiers are performing over all the domains of the testing set. Parametric evaluation of classifier performance allows for that more fine grained look at the classifiers behaviours.

2.2.2 Parametric evaluation

Some machine learning classifiers like random forest, neural networks, SVM, logistic regression classifier, etc output a set of n predictions for each example corresponding to their confidence that each example belongs to each of the n classes. Once normalized, these scores are often used as probability values of belonging to each of the n classes of a certain classification task.

By changing the value of a treshold parameter through all the possible values of probabilities obtainable with the machine learning classifier, one can monitor the efficiency of the prediction for each examples: from the most obvious to classify to the least obvious. In fine, parametric evaluation can be seen as an evaluation of all the possible confusion matrices obtainable given a prediction vector.

This parametric evaluation allows for a fine grained study of the classifier prediction. For example: two classifiers that have the same value of F1-score (see section 2.2.1 on a given testing set can have very different detection behaviours: One can be good at detecting obvious fraud but decreases in efficiency very fast, whereas the other can be not that good at detecting obvious fraud but more stable in its detection. The priorities are afterwards to be set for the best classifier with respect to the production needs.

When doing a parametric evaluation, two curves are mostly plotted: a ROC (Receiving Operating Characteristic) curve, or a precision-recall curve (see figure 2.5).

- The ROC curve axis are: true positive rate for the y-axis and false positive rate for the x-axis. Both axis are proportion of the class they are representing: positive for the true positive rate and negative for the false positive rate. Therefore, the ROC curve is unsensitive to unbalanced datasets: differences in class effectives won't affect the overall shape of the curve.

- The Precision-Recall curve axis are the precision for the y-axis and the recall for the x-axis. As for the axis of the ROC curve, the recall is a ratio of the positive class: it represents the proportion of example of the positive class. However, the precision is based on both positive class (TP) and negative

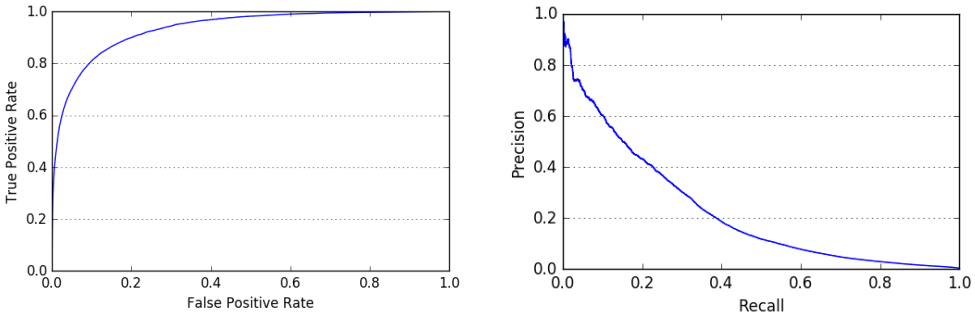


Figure 2.5: ROC curve (left) and precision-recall (right) curve evaluations for e-commerce fraud detection

class (FP) effectives. Therefore, a strong class imbalance (such as credit card fraud detection imbalance) would affect the precision values since the classification is harder. The precision recall curve is more punitive than ROC curve and consequently more challenging to improve: in a context of class imbalance, the pecision value is closer to 0 and therefore the precision recall curve decreases. Besides, we experienced that the precision recall curve is easier to read for industrial partners.

The parametric curves are not scores by themselves. In order to extract a metric from this type of curve for predictions comparison, the area under the curve is calculated:

$$\int_{treshold=t_{min}}^{t_{max}} f(x)dx$$

According to domain application needs, the area under a portion of the curve can also be calculated. For example, for credit card fraud detection evaluation, the need to calculate the AUC for values of recall below 0.2 was discussed since the experts can only investigate a limited number of alerts.

2.3 Fraud detection research questions and contributions

2.3.1 Fraud detection research questions

Credit card fraud detection is a crucial asset for ensuring customer trust and saving money by preventing fraudulent loss. Moreover, credit card fraud detection presents several characteristics that make it a challenging task and an interesting research question.

Skewed class distributions Most transactions are legitimate and only a very small fraction of the transaction has been issued by fraudster. In spring 2015 in Belgium, the observed fraction of fraudulent transaction is 0.04% for the face-to-face (card present) transactions and 0.34% for the e-commerce (card not present) transactions (see paragraph 2.1.1). Some algorithms such as Support Vector Machines for example suffer a lot from class imbalance and can be completely unable to learn the minoity class specific patterns ([Wu and Chang, 2003], [Yan et al., 2003]). Solutions have been presented in the literature in order to either adapt the algorithms (cost based methods for example) to imbalancy or solve the imbalancy in the data (sampling based methods).

Dataset shift Fraudsters tactics evolve over time. Some fraud strategies become out of date because of the constant cat and mouse play between fraudsters and experts, and new technologies show weaknesses that can be exploited by fraudster. Moreover, cardholders behaviours change over seasons. A given consumer may use his credit card at different merchants in winter than in summer, or buy different things to these merchants. Also, the purchasing behaviour has changed over years due to inflation and economic fluctuations and what fraud expert knew about the cardholders habits may not hold anymore. The combination of the three aforementioned factors: respectively the concept drift (when $p(y|x)$ changes), the seasonal covariate shift (when $p(x)$ changes) and the absolute covariate shift (when $p(x)$ changes) makes typical algorithms become out of date or irrelevant. The dataset on which these algorithms are tested doesn't correspond to the dataset they were trained on. These weaknesses create the need for algorithms that are designed in order to be updated regularly or ways to detect when an algorithm shows decreasing efficiency for the task it has been trained for and needs to be updated. Online learning schemes are very useful in order to prevent concept drift and covariate shift to affect the fraud detection systems.

Feature engineering Credit card transactions are represented as vectors of continuous, categorical and binary features that characterize the card-holder, the transaction and the terminal. The card-holder is characterized by a unique card-holder ID, its age, its gender, etc. The transaction is characterized by variables like the date-time, the amount and other confidential features. The terminal is characterized by a unique terminal-ID, a merchant category code, a country. However, most learning algorithms can't handle categorical feature without a pre-processing step in order to transform them into numerical features ([Kotsiantis, 2007]). Additionnaly, specific feature engineering strategies for credit card detection exist. These strategies either aim to make the features more significant for the classifier or to create additional features in order to enrich transaction data.

Sequence modeling A major issue of typical credit card transaction dataset is that the feature set describing a credit card transaction usually ignores detailed sequential information. Therefore, the predictive models only use raw transactional features, such as time, amount, merchant category, etc. [Bolton and Hand, 2001] showed the necessity to use features describing the history of the transaction. Indeed, credit card transactions of the same cardholder (or the same terminal) can be grouped together and compared to each other as highlighted in section 2.1.1.

Interpretability Aside from the aforementioned research questions, interpretability of the machine learning algorithms decision is a crucial concern for industrials who are the targeted users of applied data science research work such as credit card fraud detection. Research has been done in order to understand better on which heuristics model decisions are based ([Pastor and Baralis, 2019] for example). However some models remain black boxes.

2.3.2 Contributions

Solutions for integrating sequential information in the feature set exist in the literature. The predominant one consists in creating a set of features which are descriptive statistics obtained by aggregating the sequences of transactions of the card-holders (sum of amount, count of transactions, etc..) (see section 5.1).

We used this method as a benchmark feature engineering method for credit card fraud detection. However, this feature engineering strategy raised several research questions. First of all, we assumed that these descriptive statistics can't fully describe the sequential properties of fraud and genuine patterns and that modelling the sequences of transactions could be beneficial for fraud detection. Moreover the creation of these aggregated features is guided by expert knowledge whereas sequences modelling could be automated thanks to the class labels available for past transactions. Finally, the aggregated features are point estimates that may be complemented by a multi-perspective description of the transaction context (especially from the point of view of the seller).

The main focus of this work was to propose a multi-perspective HMM-based automated feature engineering strategy in order to incorporate a broad spectrum of sequential information in the transaction feature set. In fact, we model the genuine and fraudulent behaviours of the merchants and the card-holders according to two univariate features: the timing and the amount of the transactions. Moreover, the HMM-based features are created in a supervised way and therefore lower the need of expert knowledge for the creation of the fraud detection system.

Our multi perspectives HMM-based approach partially answers some of the aforementioned research questions. First, we studied how to model sequences in order to create new features. We compared solutions and chose the hidden markov model (HMM) described in section 5.2. Furthermore, we chose to use machine learning algorithms whose decisions is reachable and partly interpretable: random forests allow the user to know on which features the prediction relies the most. Hidden Markov models also contains very few parameters. It enables a compression of the behaviours observed along with an easier understanding of which behaviours are modelled.

Experiments conducted on a large real world credit card transaction dataset (46 million transactions from belgium card-holders between March and May 2015) have shown that the proposed HMM-based feature engineering allows for an increase in the detection of fraudulent transactions when combined with the state of the art expert based feature engineering strategy for credit card fraud detection. The proposed HMM-based feature engineering strategy shouldn't replace the transaction aggregation strategy since the best detection is achieved when combining these feature engineering strategies.

Chapter 3

Related Work

In this chapter we will detail the state of the art in the fields of credit card fraud detection and sequential anomaly detection. We will also describe the solutions proposed to several problems inherent to credit card fraud detection. More precisely we will show how the research questions introduced in chapter 2 are answered in the litterature.

- The approaches to answer skewed class distribution are split between data based strategies which aim to reduce class imbalance and model based strategies that aim to make the machine learning algorithm robust to class imbalance. They are presented in section 3.1.
- Different feature engineering, feature selection and feature creation not necessarily specific to credit card fraud detection are presented in section 3.2.
- Solutions for sequence modelling in the context of anomaly detection are presented in section 3.3. However hidden Markov model aren't described thoroughly in this section since they are detailed in depth in chapter 5.
- The related work chapter ends with a description of dataset shift along with the approaches that exist in order to characterize it. Afterwards, several strategies to increase the robustness of machine learning classifiers to dataset shift are presented.

3.1 Learning to classify unbalanced datasets

Typical fraud detection tasks present a strong imbalance in the representations of the classes: there is usually a majority genuine class that outweighs the minority fraudulent class [Ali et al., 2019].

Most learning algorithms are not adapted to a strong imbalance in the representations of classes in the dataset. Therefore, learning from unbalanced dataset is a challenging research question that needs to be considered([Batista et al., 2000]). [Weiss and Provost, 2001] and [Estabrooks et al., 2004] observed that an unbalanced dataset leads to a decrease in the performance of standard learning algorithms such as SVM or random forest. In fact, class imbalance have been shown to produce a negative effect on the performance of a wide variety of classifiers ([Chawla, 2005]). [Japkowicz and Stephen, 2002] have shown that skewed class distributions affects negatively the performance of decision trees and neural networks. [Visa and Ralescu, 2005] drew the same conclusion for neural networks. [Kubat and Matwin, 1997], [Mani and Zhang, 2003] and [Batista et al., 2004] showed the weakness of k Nearest Neighbors algorithms to imbalanced datasets. [Yan et al., 2003] and [Wu and Chang, 2003] showed that SVM were also weak to skewed class distributions. Overall, most of the classifiers suffer from class imbalance, some more than others.

The strategies used to tackle this issue operate at two different levels: some aim to solve the imbalance of the dataset beforehand whereas others want to adapt the learning algorithm to make it robust to an unbalanced classification task ([Chawla et al., 2004], [Pozzolo, 2015]).

The data level strategies are called like that since they happen during the preprocessing time, before any learning algorithm is applied. They consist in rebalancing the dataset in order to compensate the structural imbalance. The algorithmic level strategies involve methods where the algorithms are designed especially to deal with an unbalanced task and methods where the classification costs of the algorithm are adapted in order to reflect the priorities of the unbalanced classification task. The latter are called cost sensitive classifiers ([Elkan, 2001]).

The sampling based solutions to handle imbalance will be described in section 3.1.1. The model based methods will be briefly highlighted in section 3.1.3

3.1.1 Sampling methods

Sampling methods consist in reducing the class imbalance by changing the class effectives in the dataset. In this section we will first present two naive methods that reduce class imbalance in the dataset by adjusting class ratios: undersampling and oversampling. Afterwards we will introduce the Synthetic Minority Oversampling Technique strategy (SMOTE) that aims to create new examples of the minority class [Chawla et al., 2002]. The different strategies are illustrated in figure 3.1.

Undersampling consists in rebalancing dataset by taking less examples from the majority class in order to match the effectives of the minority class ([Drummond and Holte, 2003]). The assumption made with undersampling is that the majority class contains redundant information that can be removed. The main issue with undersampling is that, when the imbalance is too pronounced, too many examples from the majority class need to be taken away. This may cause a decrease in the performance of the algorithm due to a lack of data. It is worth mentioning that undersampling speeds up the learning phase, which makes it an interesting choice when in presence of an unbalanced dataset.

Oversampling consists in rebalancing the dataset by repeating random examples of the minority class ([Drummond and Holte, 2003]). The main issue of oversampling is that repeating some examples doesn't add information and may lead to an overfitting of the learning algorithms. Besides, oversampling increases the learning time since it makes the train-set artificially bigger.

In 2002, [Chawla et al., 2002] introduce SMOTE, a strategy to reduce class imbalance by creating new examples in the neighbourhood of the minority class. The creation of new minority class example is done by averaging neighboring examples of the minority class. For categorical features, the averaging consists in a majority voting for the category: the category mostly represented in the neighbourhood becomes the category of the corresponding categorical feature of the newly created example. They show an increase in the performance of classifier when using this rebalancing strategy. However SMOTE does not consider the label of neighboring examples when creating examples from the minority class. This can lead to an overlap between minority and majority classes. ADASYN [He et al., 2008] and Borderline-SMOTE [Han et al., 2005] tackle this issue by taking into account only neighboring examples from the minority class.

3.1.2 Cost based method

The metrics obtained using the confusion matrix (see section 2.2.1) usually gives the same weights to the true positive, false positive, true negative and false negative. However, in presence of class imbalance, this can cause issue. For example, in the case where the negative class is representing 10% of the dataset and the positive class 90% of the dataset, the confusion matrix of a very bad classifier which says that every example is positive without having any insight on the patterns representing the positive and negative classes would be the one shown in table 3.1. The corresponding accuracy would be: $ACC = \frac{90+0}{90+10+0+0} = 0.9$. Because of the class imbalance, this very bad classifier is measured to predict the good class 90% of the times even if it doesn't contain valuable information.

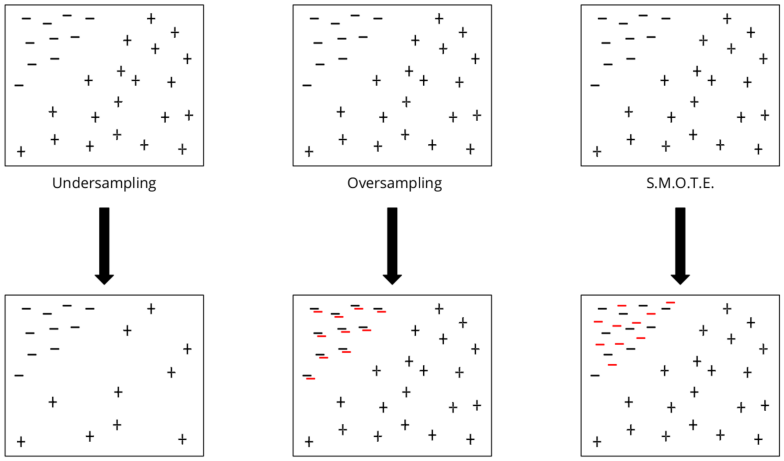


Figure 3.1: Sampling strategy for unbalanced classification. *(The + symbols refer to the majority class whereas the − symbols refer to the minority class. The symbols colored in red are newly created symbols by the oversampling strategy or by the SMOTE strategy([Chawla et al., 2002]))*

Table 3.1: 90/10 imbalance confusion matrix

	predicted positive	predicted negative
actual positive	90	0
actual negative	10	0

In order to solve this issue, the misclassification costs of the false positives (**FP**) and false negatives (**FN**) can be used in order to replace the value of the false negatives and false positives in the confusion matrix with respect to the ideal ratio of majority/minority examples. The $\frac{majority}{minority}$ ratio would be equal to $\frac{falsenegativecost}{falsepositivecost}$ [Zadrozny et al., 2003]. The adjusted confusion matrices obtained when testing models will therefore reflect correctly the importance of misclassifying a positive and a negative example.

3.1.3 Model based methods

We introduced in section 2.1.2 ensemble based methods since they are common methods for the credit card fraud detection application domain. Some solutions were published in order to strenghten ensemble based methods when in presence class imbalance. EasyEnsemble ([Liu et al., 2009]), UnderBagging ([Wang et al., 2009]) and SMOTEboost ([Chawla et al., 2003]) are ensemble based method that are adapted by design to skewed class distributions. SMOTEboost ([Chawla et al., 2003]) combines boosting with SMOTE in order to provide balanced datasets to each boosting weak learner. Similarly, EasyEnsemble ([Liu et al.,

2009]) use undersampling in order to create different balanced datasets for the boosting learners. In UnderBagging ([Wang et al., 2009]) undersampled datasets are used to train several learners whose predictions are aggretated with bagging.

An other leverage for dealing with class imbalance is to adapt the metrics used to evaluate the models and the costs used to train some of them in order to counterbalance class imbalance. Indeed, some real world imbalanced classification tasks such as credit card fraud detection give more importance to correctly identify the element of the minority class than those of the majority class. In the case of cost sensitive learners, the misclassification cost can be adapted in order to reflect this priority. [Shirazi and Vasconcelos, 2010] present cost sensitive SVMs. [Kukar and Kononenko, 1998] introduced earlier the possibility of cost adaptation for neural networks. Adapting the misclassification cost for imbalanced problem has also been studied for boosting methods ([Sun et al., 2007], [Fan et al., 1999]).

In the end, there is a variety of methods to deal with class imbalance. In section 4, different sampling methods are compared in order to find which one is the more appropriate for our unbalanced credit card fraud detection problem.

3.2 Feature engineering for credit card fraud detection

As stated in chapter 2, transactions are represented by vectors of continuous, categorical and binary features that characterize the card-holder, the transaction and the terminal. The card-holder is characterized by a unique card-holder ID, its age, its gender, etc. The transaction is characterized by variables like the date-time, the amount and other confidential features. The terminal is characterized by a unique terminal-ID, a merchant category code, a country.

However, most learning algorithms can't handle categorical features without a preprocessing step in order to transform them into numerical features ([Kotsiantis, 2007]). In section 3.2.1 we will discuss the different solutions that were considered in order to encode the categorical features. We will also present some papers that describe feature selection methods and explain which features we chose to use for our credit card fraud detection task.

Also, credit card fraud detection can be improved by providing additional information to the classifier. For example, the information usually contained in the features of the credit card transactions doesn't reflect the history of the transactions. In section 3.2.2 we will describe some state of the art solutions in order to enhance the information contained in the credit card transaction by

creating new features.

3.2.1 Feature encoding and feature selection

Feature encoding

Several options are presented for encoding categorical features in the literature. The two most common options are label encoding and one-hot encoding ([Kotsiantis, 2007].

Label encoding is an encoding strategy where the categorical feature with n categories is transformed into a numerical feature taking n different numerical values. The asset of label encoding is that it is light memory wise since you don't increase the shape of the dataset matrices. However, with label encoding the categories become ordered after their transformation into numbers. This can cause issues for some classifiers like SVMs, neural networks or logistic regression classifiers that aim to calculate a distance between examples. On the other hand tree based classifiers aren't affected by the order of the categories due to label encoding since their classification process consists in splitting the dataset (see section 2.1.3).

One hot encoding is an encoding strategy where the categorical feature with n categories is transformed into $n-1$ binary features describing the values of the categorical feature. This method is appropriate for distance based classifiers since it doesn't order the categories: all of them are equidistant whereas with label encoding there are some categories that would be falsely considered closer than others.

An other elegant solution that maintains an appropriate distance measure between categories of a categorical features is class embeddings. Moreover, it is significantly less memory intensive than one-hot encoding. It consists in mapping the different categories of a feature to a k dimensional space (with $k \ll n_{categories}$) and to adjust the mapping with a gradient descent process. [Guo and Berkhahn, 2016] used embeddings as a way to encode categorical features in a supervised way. The embeddings maps are equivalent to an extra layer of neurons on top of the pre one-hot encoded feature set. They are adapted to the classification task thanks to a gradient descent process. Recently [Russac et al., 2018] (in collaboration with the chair of data science of the university of Passau) used Word2Vec embeddings in order to replace one-hot encoding for a distance based classifier (logistic regression) in a credit card fraud detection task.

[Carcillo et al., 2018] and [Pozzolo, 2015] showed that frequency encoding can

be useful in the case of an imbalanced classification task like credit card fraud detection. With frequency encoding, the value of the category is replaced by the ratio $\frac{MinorityClass}{MajorityClass}$ (the percentage of fraud in this category for our application).

Feature selection

Typical credit card transactions have a big number of attributes Some of the attributes like the city of the card-holder may be not meaningful for the classifiers or lead to overfitting issues because they present too many categories or are too sparse. Also, some features may contain the same information twice (for example the ZIP code contains the value of the county) or can be highly correlated like the age of the card-holder and the mean amount of its transactions for example.

In this context, feature selection can be useful in order to purify the feature set, decrease the duplicates in informations and keep only relevant information for the machine learning task. Moreover, having a smaller number of features and less correlated features containing duplicate information thanks to feature selection would lead to a better understanding of the classifiers decision. Besides, feature selection may increase the speed of classifiers (smaller feature set and datasets) and their performances (less overfitting) ([Bhattacharyya et al., 2011]).

[Noghani and Moattar, 2015] proposed an iterative approach for feature selection in a credit card fraud detection application. They add one by one the features that improve the most the performance of a random forest classifier.

Leopold Fossi and Gabriele Gianini used the Shapley value [Shapley, 1953] in order to do feature selection for credit card fraud detection [Fossi and Gianini, 2019]. Shapley value reflects the contribution of a player to the team: a good player may not add too much for a team because his strengths are already brought by an other player whereas a not so good player may fulfill the needs of a team and benefit more to it.

Feature selection may become mandatory when the data is enriched with a lot of new informative features. [Fawcett and Provost, 1997] proposed to describe the evolution of relevant credit card fraud detection features for different hours of the day. They stated afterwards the need of a feature selection process in order to decrease the total number of features.

Missing values strategies

Missing values is an oftentimes met issue in machine learning task [Kotsiantis, 2007]. Some datasets may contain missing values, either because some informa-

tion wasn't provided or because the values are structurally missing: for example informations about internet browser used for the purchase may be available in e-commerce credit card transactions dataset but are missing for face-to-face transactions.

As stated by [Acuna and Rodriguez, 2004] there are 3 main approaches to solve missing value issue:

- The most straightforward way to handle missing values is to delete them. One can choose to delete the rows containing the missing values if these aren't a too large part of the dataset. Besides, it is also possible to choose to delete the features concerned if they are too sparse. It becomes a feature selection problem (see paragraph 3.2.1).

- One can also keep the rows containing missing values and fill the missing values with replacement values like the mean, the median, or a default categorie (-9999 for a card-holder age for example). [Troyanskaya et al., 2001] compared these approaches in the context of DNA microarrays.

- Finally, a more costly and elegant solution can be to infer the missing value of the using a model. [Hyunsoo et al., 2005] used k-nearest neighbors for missing value imputation for a genomics application. [Oba et al., 2003] chose to use bayesian principal component analysis for missing value imputation also in a DNA microarrays dataset. [Stekhoven and Buhlmann, 2011] proposed MissForest: a random forest approach for missing value imputation that leverages the random feature choice for trees in order to estimate out-of-bag errors tree wise. Imputations methods are relative to the particular problem: in presence of a highly categorical dataset, the user may want to use tree based classifiers for missing value imputation, whereas principal component analysis is less computationally expensive and totally reasonable for a numerical dataset.

In the end missing values is a very current issue in machine learning and the solutions proposed to handle them are simple and efficient.

3.2.2 Feature creation for credit card fraud detection

[Bolton and Hand, 2002] highlighted the fact that even if the transactions contain a lot of attributes, there is a need for creating new attributes in order to describe better the transactions. They showed for example that a first time credit card user may not have the same buying behaviour as a long time credit card user and adviced to create features to bring historical knowledge to the classifiers.

History based features

Several strategies have been explored in order to introduce history based information to the feature set.

Traditionally, the feature engineering for credit card fraud detection uses the fundamentals of RFM (Recency - Frequency - Monetary). Recency can be characterized by the time-delta feature reflecting the time elapsed between two consecutive transactions, the Frequency can be characterized by the number of transactions in a period of time, and the Monetary aspect describes the amount spent by the card-holder. This RFM fundamentals are described in [Blattberg et al., 2008] and used as state of the art by [Vlasselaer et al., 2015], [de Fortuny et al., 2014].

[Fu et al., 2016] introduced a new feature called trading entropy that aims to quantify: "How surprising is the recent amount of money spent for this card-holder with this type of merchant". The proportion of the i-th merchant type in the last k transactions is pi:

$$pi = \frac{TotalAmountForMerchantType_i}{TotalAmountOfLastKTransactions} \tag{3.1}$$

The feature proposed by [Fu et al., 2016] is the variation of entropy with the addition of the incoming transactions to the amount history of the considered card-holder:

$$TradingEntropyT = \sum_{i=0}^{K-1} p_i * log(p_i) - \sum_{i=1}^{K} p_i * log(p_i) \tag{3.2}$$

Earlier, [Bolton and Hand, 2001] adviced similarly to refer to the past transactions of the card-holder for outlier detection. Indeed, break point analysis aims to quantify how much a global outlier raised is a real outlier for the considered card-holder.

[Whitrow et al., 2008] proposed a transaction aggregation strategy to create descriptive features containing information about the past behaviour of the card-holder over a certain period of time. They showed a 28% increase in the detection of fraudulent transactions by using these aggregated features with a random forest as the learning algorithm.

The aggregations are calculated for a transaction based on the card-holder preceding transactions. Given the incoming transaction x_i, the previous transactions of the card-holder are selected with respect to *particular* features. The set of transactions S_i selected is for example the transactions done with merchants

from the same country or with merchants of the same type. The value of the new features are calculated from S_i via aggregation functions:

$$sum_i = \sum_{x_{amt} \in S_i} x_{amt} \tag{3.3}$$

$$count_i = |S_i| \tag{3.4}$$

[Jha et al., 2012] applied Whitrow's transaction aggregation strategy to logistic regression in a real world credit card transactions dataset. [Bahnsen et al., 2016] also used Whitrow's transaction aggregation strategy as the state of the art for their work (described in the next section) with random forest, Logistic Regression and Bayes minimum Risk model. Wihtrow's transaction aggregation strategy reflects an information similar to the RFM strategy except that for RFM the history isn't selected with respect to some categorical features such as merchant type or merchant country.

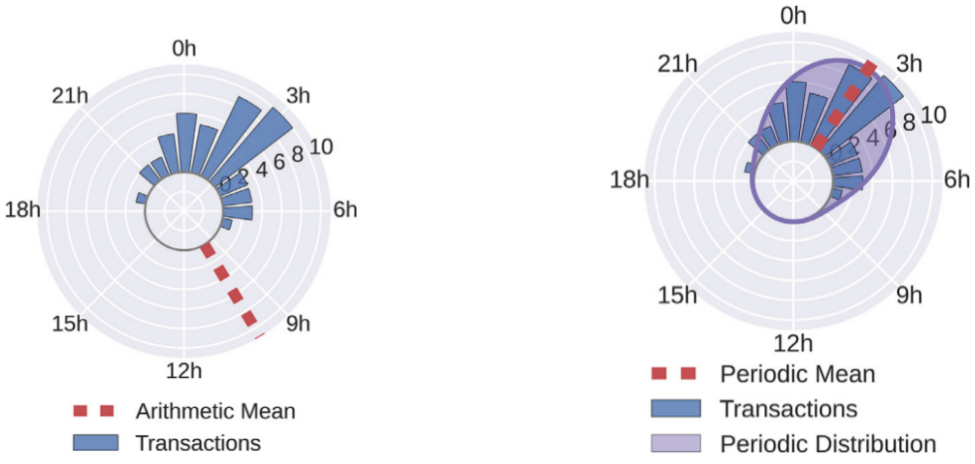
We have shown in section 5 that the features based on transaction aggregation strategy are very informative features for credit card fraud detection. Therefore, the main work of this PhD was to propose a feature engineering strategy that overcomes the limits of the transactions aggregation strategy of [Whitrow et al., 2008].

Other feature creation approaches

In order to enrich the transaction information, [Vlasselaer et al., 2015] propose APATE, a framework that aims to create network based features. The network based features are constructed by propagating the label of the edges (i.e. transactions) in order to infer a time dependent suspiciousness score for each node (card-holder and merchants) of the bipartite graph of the transactions already introduced in section 2.1.1.

[Bahnsen et al., 2015], [Bahnsen et al., 2016] highlighted that the time features have a periodic behavior that needs to be taken into account during the preprocessing phase. The time features are not usual numerical features as shown in figure 3.2 (for example the arithmetic mean doesn't apply well to time features).

They proposed to use the Von Mises distribution ([Best and Fisher, 1979]) in order to reflect better the periodicity of the time features when calculating time deviation values (see figure 3.2).



Uniform distribution

Von Mises distribution

Figure 3.2: Time of the transaction on a 24h clock modeled with the uniform distribution (*left*) or the Von-Mises distribution (*right*). (from [Bahnsen et al., 2016])

3.3 Sequence modeling for anomaly detection

Sequence modelling is a major machine learning research question. In this section we will split the field with respect to two main approaches: the generative approach that aims to learn the joint distribution of the events in a sequences and the discriminative approach that aims to find the labels of the events in a sequence.

Afterwards, we will highlight how these approaches are integrated in anomaly detection tasks.

3.3.1 Sequence modelling

Graphical models

Hidden Markov models are generative models especially built for sequence modelling. They are briefly introduced here and described more in depth in section 5.2.2

They comprise two matrices (see figure 3.3):

- The *transition matrix* describes the succession of the hidden states. It reflects a multinomial distribution from all hidden states at time t to all hidden states at time $t+1$. The hidden states obey to the Markov property. Indeed, the hidden state at the time $t+1$ only depends on the hidden state at the time t: There is no memory in the distribution of successive hidden

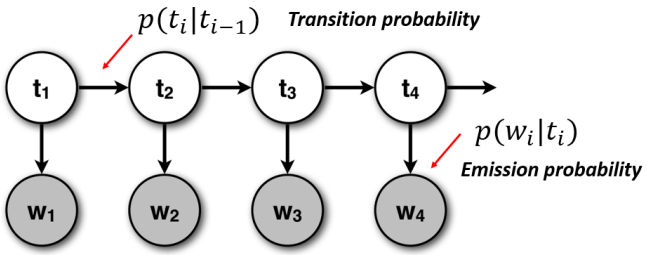


Figure 3.3: Hidden Markov model architecture. (from [Lucas et al., 2019a])

states, only conditional probability given the immediate previous hidden state. Formally, it means that conditional distributions between hidden states t in a sequence of size n can be simplified as following:

$$p(t_k|t_{k-1}, t_{k-2}, .., t_1, w_{k-1}, w_{k-2}, w_1) = p(t_k|t_{k-1}) \; k \in \{1, .., n\}$$

- The *emission matrix* describes the conditional distribution of the observed variable w_i given the hidden states t_i. The probability distribution changes with respect to the observed variable properties. For example it can be considered multinomial for categorical observed variables or gaussian for continuous observed variables.

The transition and emission conditional probability matrices of the hidden Markov models are usually initialized with a random value and optimized by an iterative Expectation-Maximisation algorithm known as the Baum-Welch algorithm ([Baum, 1972] [Rabiner and Juang, 1991]). Similarly to a training process, this optimisation process will adapt the transition and emission matrices in order to reflect the distribution of a set of observed (training) sequences. EM optimization of a model with latent (hidden) parameters consists in:

expectation Finding the latent states distributions that correspond the most to the sequences of observed data. This is usually done with the help of the Viterbi algorithm which recursively leverages the Markov property in order to simplify the calculations of the conditional probabilities (also referred to as forward-backward algorithm [Viterbi, 1967]).

maximisation Maximising the correspondence between the latent distribution inferred during the expectation step and the parameters of the transition and emission matrices by adjusting the parameters.

The expectation and maximisation steps are repeated until convergence of the graphical model to the observed data. The convergence can be monitored via the evolution of the likelihood that the set of observed sequences have been generated

by the model. This likelihood increases over the iterations until it reaches a ceiling when the hyper parameters ruling the architecture of the generative model don't allow it to fit more to the set of observed sequences.

Even if HMMs are an elegant graph-based model, they present one drawback: their structure often doesn't reflect the true process producing the observed data. This problem comes mostly from the Markov property: any relationship between two non consecutive hidden states ($y1$ and $y4$ in figure 3.4 for example) must be communicated through the intermediate hidden states. A first-order Markov model like HMM (where $p(y_t)$ only depends on y_{t-1}) is less able to capture these longer term relationships.

Several architectures have been tried in order to overcome the modelling weaknesses of HMMs: maximum entropy markov models ([McCallum et al., 2000]), input-output HMMs ([Bengio and Frasconi, 1995]), and conditional random fields ([Lafferty et al., 2001]) (see figure 3.4). However, as represented by the directions of the arrows in figure 3.4, all of these architectures are discriminant models that represent the conditional distribution $p(y|x)$ rather than $p(x, y)$. Discriminant models are usually more efficient than generative model since the conditional distribution $p(y|x)$ is simpler than the joint distribution $p(x, y)$: the joint distribution $p(x, y)$ contains the conditional distribution [1]. On the other hand, the joint distribution also contains the information of the data distribution x. One may want to choose HMM especially for their ability to model all the data distribution $p(x, y)$ and not only the conditional distribution of the target variable $p(y|x)$.

Recurrent neural network approaches

Recurrent neural networks (RNNs) are a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. These connections allow to model temporal dynamic behaviors.

Recurrent neural networks can be considered as an hidden state based model where the inner layers reproduce the hidden state based architectures. Furthermore, the connectivity between consecutive hidden states (or nodes) also follows the Markov property ($p(y_t)$ only depends on y_{t-1}). RNNs are discriminant models that aims to predict the label of an event given a past sequence of events. They are detailed extensively in [Graves, 2012].

According to [Bengio et al., 1994] Recurrent network is a powerful model

[1] since $p(x, y) = p(y|x) * p(x)$

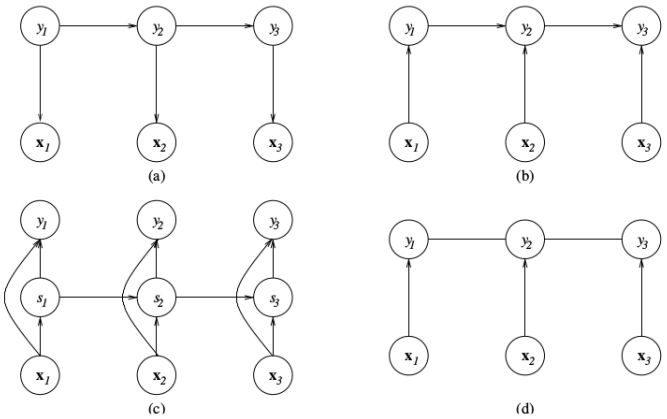


Figure 3.4: Graphical model architectures. (*(a) HMM, (b) maximum entropy markov model, (c) input-output HMM, (d) conditional random field)* (from [Dietterich, 2002])

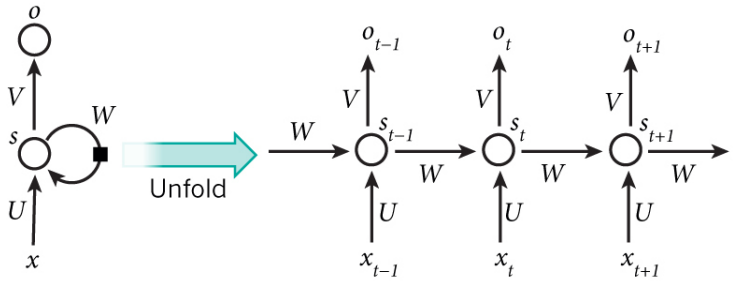


Figure 3.5: Stacking networks time-wise for backpropagation through time

that presents difficulties for training. In order to train a RNN, Backpropagation through time is often used (BPTT), however it causes *vanishing and exploding gradient* problem because BPTT consists in representing the network as a deep multi layer network that stacks the initial RNN as many time as there are time steps (see figure 3.5). The virtual network on which backpropagation is applied ([Rumelhart et al., 1986]) becomes very deep.

[Mikolov et al., 2011] showed that one solution to *exploding gradient* can be to clip the elements of the gradient that exceed a fixed threshold.

[Hochreiter and Schmidhuber, 1997] highlighted that it is possible to overcome the *vanishing and exploding gradient* problem with the introduction of memory cells. The network structure that includes memory cells is called Long Short-Term Memory Network [Bayer, 2015]. The LSTM were shown to be able to learn long term dependencies in sequences with good success for sequential tasks such as speech recognition ([Graves and Jaitly, 2014]

[Jurgovsky et al., 2018] compared LSTMs to tree based classifiers for a credit card fraud detection task. They showed an increase in the detection of fraudulent transactions by using sequential classifiers (even over the feature aggregation strategy proposed by [Whitrow et al., 2008] for face-to-face transactions).

3.3.2 Sequence modelling for sequential anomaly detection

Sequential anomaly detection aims to raise an anomaly score for events, subsequence of events or whole sequence with respect to a genuine distribution.

For sequential tasks, multiple definitions of anomalies exist that need to be handled differently ([Chandola et al., 2012]):

- An event of a sequence may be anomalous. For example, when a transaction is fraudulent within the sequence of transactions of a given card-holder.
- A subsequence may be anomalous. For example, if there is a delay between the credit card theft and its reporting, the transactions issued with the credit card are not made by the card-holder but by the fraudster and must be identified as anomalous.
- A sequence may be anomalous with respect to a set of non anomalous sequences. For example, a sentence in french will be anomalous in a set of english sentences. Or given a database of protein sequences, some sequences may be anomalous.

One simple way to raise an anomaly score is counting the likelihood of the event or fixed length window of events in the genuine set of events.

$$L_i = \frac{\#subsequence(i)}{\#subsequences} \tag{3.5}$$

[Forrest et al., 1999] proposed the t-STIDE technique (treshold based sequence time delay embedding) in order to raise anomaly scores for sequences. It consists in computing the ratio of sliding windows that have a likelihood (see equation 3.5) below a certain treshold λ. The anomaly score of the test sequence is proportional to the number of anomalous windows in the test sequence. The corresponding formula is:

$$A_{Sq} = \frac{|i : L(w_i) < \lambda, 1 \leq i \leq t|}{t} \tag{3.6}$$

This kind of window based counting can also be used as anomaly signal to calculate a 2nd order anomaly score. For example, [Hofmeyr et al., 1998] proposed to compute the average of the strength of the anomaly signal over a set of windows

that cover the considered event or subsequence.

$$A(Sq) = \frac{1}{N} \sum_{i=1}^{N} A(w_i) \tag{3.7}$$

In addition to counting methods for calculating anomaly scores, machine learning based models can also be used in order to raise these anomaly scores.

Hidden Markov models (HMMs) are the most used model in the field of anomaly detection for discrete sequences according to [Chandola et al., 2012]. [Gornitz et al., 2015] used HMMs in order to raise anomaly score for finding genes in the genome of prokaryotes. [Rashid et al., 2016] used HMMs in order to model genuine behaviour for the task of detecting malicious insiders in the sequence of logs.

[Srivastava et al., 2008] used multinomial HMMs in the context of credit card fraud detection. They modelled the sequence of amount of users as a categorical feature with 3 values: "low amount" for the transactions with amounts less than 50 €, "mid amount" for transactions with amounts between 50 and 200 € and "big amount" for transactions with amounts bigger than 200 €. The anomaly score raised is the inverse of the likelihood of the recent amounts to have been generated by an HMM trained on a set of genuine sequences. They generated a credit card transactions dataset in order to assess the qualities of their strategies. [Dhok, 2012] used a similar approach for a real world credit card transaction dataset, and [Robinson and Aria, 2018] extended that with success for a prepaid credit card fraud detection task.

[Dorj et al., 2013] also used HMMs in an anomaly detection task in electronic systems, but they modelled an abnormal sequence in addition to a genuine sequence. This is interesting since assessing the abnormality of outliers detected decreases the number of false positive. You not only raise outliers, but you also confirm that these outliers belong to the abnormal class.

HMMs for sequential anomaly detection have mostly been used in order to raise an anomaly score using the likelihood of the sequence to have been generated by the HMM. However, recurrent neural network (RNN) approaches and more precisely long short term memory networks were more varied. Similarly to the precited HMM approaches, LSTM were used in order to raise anomaly scores by [Bontemps et al., 2016] for an intrusion detection task. The anomaly scores of a window of events is the distance between this window of events and the window of events that would have been generated by the LSTM for the corresponding user. Since LSTM are discriminative models, a custom distance has been created

by the authors for this generative task. [Malhotra et al., 2015] modelled the genuine distribution of sequence based dataset with LSTM in order to afterwards raise an anomaly score for electrocardiograms, space shuttle valve time series and power demand datasets. Alternatively, [Ergen et al., 2017] used LSTM to extend various length sequences in order to obtain sequences of equal length for raising an anomaly score afterwards with the help of One Class SVM.

3.4 Dataset shift detection and adaptation

Many machine learning tasks make the assumpion that the random variables in the data are i.i.d. (independent and identically distributed) among the train and the test sets. That property ensures that the decision function learned by the classifier on the train set is valid on the testing set. However in some case the i.i.d. hypothesis isn't valid: there are changes in the data distribution between the training and testing set. These changes of the data distribution can decrease drastically the performances of the classifiers ([Pozzolo et al., 2014], [Abdallah et al., 2016], [Alaiz-Rodriguez and Japkowicz, 2008]) and need to be taken into account.

The term "dataset shift" is used in order to include the phenomenons where the i.i.d. hypothesis isn't valid. It is defined by [Moreno-Torres et al., 2012] as "cases where the joint distribution of inputs and outputs differs between training and test stage"

The two most frequent causes of dataset shift are non stationary environments and sample selection bias. Sample selection bias refers to the fact that the training data is gathered differently from the data of the testing set. For example, data for a driving behaviour modelling task can be collected with the help of students that play a driving simulator game. This can cause a sample selection bias because the context of acquisition of the training set (artificial driving simulator) is different from the real life application (real life road safety task).

Non stationary environments can make the distribution of the data change. For example in the case of the bird sound recognition challenge of LIFECLEF [LifeCLEF, 2018], the dataset spans from January to June 2017: seasons may affect the patterns observed and the class distribution (i.e. bird species observed). Similarly, spatial changes may affect learning task.

3.4.1 Different types of dataset shift

Dataset shift is a broad word that refers to a change in the joint distribution of the target variable and the input features $p(x, y)$. However, since $p(x, y) = p(y|x) * p(x)$, the causes of the change in the joint distribution can be different. Several shifting phenomenon can be identified ([Moreno-Torres et al., 2012], [Storkey, 2009], [Gao et al., 2007], [Shimodaira, 2000]):

- *Covariate shift* is the type of shift that appears most often in the literature. It was formerly called *population drift*. It refers to the phenomenon when the conditional distribution of the classes with respect to the training data doesn't change from the train set to the test set, but the distribution of the data do. For example in the credit card fraud detection settings, the phenomenon where the buying behaviors change over seasons but the fraudulent strategies don't can be seen as *covariate shift*. It is formally defined as:

$$p_{train}(y|x) = p_{test}(y|x) \quad \text{and} \quad p_{train}(x) \neq p_{test}(x) \tag{3.8}$$

- *Prior probability shift* is an other subtype of dataset shift. It is also sometimes called *shifting priors*. *Prior probability shift* refers to a change in the class distributions: for example when the proportion of fraudulent transactions in the training set is way smaller than for the testing set. It may cause a surrepresentation or an underrepresentation of one class in the prediction of machine learning classifiers. It is formally defined as:

$$p_{train}(y|x) = p_{test}(y|x) \quad \text{and} \quad p_{train}(y) \neq p_{test}(y) \tag{3.9}$$

- *Concept shift* or more often *concept drift* refers to the phenomenon where the conditional distribution of the target classes with respect to the input variables changes. For example when fraudster adapt their strategies and the classifier is not able to detect fraud efficiently anymore. It is formally defined as:

$$p_{train}(y|x) \neq p_{test}(y|x) \quad \text{and} \quad p_{train}(x) = p_{test}(x) \tag{3.10}$$

3.4.2 Detecting dataset shift

Most of the studies passively detect dataset shift when a decrease in classifiers performance on the test set is witnessed([Pozzolo, 2015], [Gomes et al., 2017],...). In order to prevent the decrease of classifier performance, some teams aimed to estimate more realistically the expected performance of classifiers in presence of dataset shift. For example, [Webb and Ting, 2005] pointed out the weaknesses of the ROC evaluation in presence of *prior probability shift*. Earlier, [Kelly et al.,

1999] stated that the weaknesses of classifiers in presence of covariate shift and prior probability shift can be highlighted with the decrease of classifiers performance for their loan risk prediction task. Additionaly, [Sugiyama et al., 2007] discussed the issues appearing when classifer performance expectation is set using a cross validation hyperparameter optimization: cross validation expectations can't be trusted if the distribution changes between train and test set. They proposed a cost based approach by comparing the data distribution $p(x)$ in the train and test set and weighting more the data points of the train set that are similar to those of the test set for the cross validation optimization. If the data point is typical of the train and test distribution, then it is weighted more for the evaluation of the model.

[Karax et al., 2019] provided interesting results where they showed how to describe in advance the concept shift in two dataset (Airlines and Sea [2]) by looking directly at the changes in the decision function through the changes of feature importance in an adaptive random forest classifier (described in next paragraph) [Gomes et al., 2017]. They could therefore predict the direction of the dataset shift and the resulting change in the conditional distribution $p(y|x)$. Earlier, [Lane and Brodley, 1998] tried to quantify the direction of drift on a continuous one dimensional axis by looking at which examples get misclassified when concept drift happens. An optimisation of the treshold value of their classifier was managed in the decision process of their user identification for computer security task. Similarly, [Wang et al., 2003] wanted to precisely characterize concept shift by comparing the class prediction between two classifiers trained before and after a concept shift (old training data vs recent training data). The examples that present a conflict for class prediction can then be studied in order to understand which changes happen between the conditional distributions $p_{old}(y|x)$ and $p_{new}(y|x)$

3.4.3 Strategies in presence of dataset shift

Data based methods Dataset shift of various types may affect classifiers performance. [Alaiz-Rodriguez and Japkowicz, 2008] made the hypothesis that simple classifiers are more robust to population drift (covariate shift) than complex classifiers. They conclude the opposite: neural networks with 10 neurons in one hidden layer and C4.5 decision tree are more robust than linear regression and neural networks with 1 hidden neuron as hidden layer on an artificial medical dataset stating the prognosis of patients infected with the flu.

Others authors preferred kernel approaches in order to handle dataset shift.

[2] `https://moa.cms.waikato.ac.nz/datasets/`

[Gretton et al., 2009] used reproducing kernel hilbert spaces (RKHS) in order to weight instances in the training set so that the averages of the training set features correspond to those of the test set features. This kernel mean matching approach is mostly used in the context of covariate shift since it allows to make the distribution of the data in the train and test set more similar. [Bickel et al., 2007] leveraged this kernel mean matching approach for data streams. They derived the kernel mean matching approach in order to integrate it in the logistic regression classifier used and to update it over time. They highlighted the difficulties of this integration for more complex classifiers. In addition to kernel mean matching, [Klinkenberg, 2003] propose to train SVMs with an adaptive time window in order to prevent covariate shift.

Similarly to the kernel mean matching approach, [Moreno-Torres et al., 2013] proposed a method in order to achieve transfer learning between different cancer diagnosis datasets. They used genetic programming in order to create a regressor that transforms the features of one dataset to the features of an other dataset in order to be able to apply a pretrained classifier without adapting it.

Classifier based methods An other way to prevent concept drift is to update regularly the models in order to adapt them to the dataset shift.

At first, sliding windows based approaches have been explored. The idea is that by retraining them regularly using fresh data, the classifiers will stay up to date with dataset shift events. [Widmer and Kubat, 1996] used a sliding window strategy for their FLORA4 framework: they trained regularly new classifiers on recent and trusted data but stored the bad performing ones for hypothetical later usage. [Lichtenwalter and Chawla, 2009] also leveraged the sliding window strategy for imbalanced data streams with concept drift: they re-trained regularly C4,5 decision trees classifiers in order to prevent covariate shift. Moreover, they adapted the information gain calculation used when building the C4,5 trees (see section 2.1.3) by incorporating the Hellinger distance in order to increase the robustness of the C4,5 trees to class imbalance and to prior probability shift. The Hellinger distance is a symmetric and non-negative measure of similarity between two probability distributions. The equation 3.6 corresponds to the Hellinger distance between two discrete probability distributions with equal effectives $P = (p_1, ..., p_k)$ and $Q = (q_1, ..., q_k)$.

[Pozzolo, 2015] proposed a sliding window method for data streams with delayed information on the same credit card transactions dataset used in this thesis (see figure 3.7). Since the ground truth for the class of credit card transactions is delayed by a week in average, the label of the most recent transaction isn't known. They proposed to aggregate the predictions of a classifier on the old

$$d_H(P,Q) = \sqrt{\sum_{i \in V} (\sqrt{P}) - \sqrt{Q})^2} \tag{3.11}$$

Figure 3.6: Hellinger distance: mesure of distributional divergence between two probability distributions P and Q

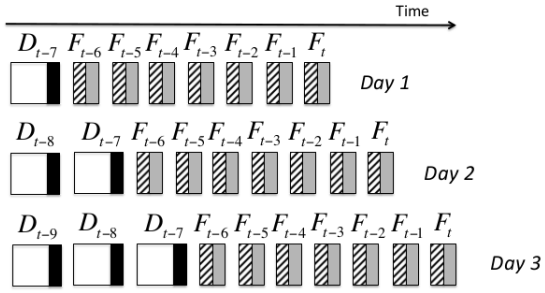


Figure 3.7: Aggregating ground truth examples with investigators feedbacks ([Pozzolo, 2015])

ground truth example with the predictions of a classifier trained on the feedbacks of the investigators on alerts raised previously by the fraud detection system. They showed that aggregating predictions provided better fraud detection than pooling all the examples of the ground truth and investigators feedback together.

Instead of preventing dataset shift by constantly re-training the classifiers, one may want to retrain them only if they prove to be lackluster. [Kolter and Maloof, 2003] proposed to train again the classifiers when their performance drop. In order to prevent missclassification for data streams, [Hoens et al., 2011] used Hellinger distance on all the features of the dataset in order to detect early a covariate shift and which features are affected by it. They would therefore be able to know which trees of a random forest classifier will be affected by the dataset shift and need to be re-trained.

More recently, authors in [Gomes et al., 2017], [Barddal and Enembreck, 2019] proposed an autonomous architecture for random forest classifier introducing a variant to Hoeffding Adaptive Tree in order to include a drift detector inside decision nodes to monitor the internal error rates of the tree, thus, continuously selecting features and building its predictive model over time. The Hoeffding Adaptive Trees (also called Very Fast Decision Trees) leverage the observation that using only a subset of examples in order to build a decision tree is sufficient to obtain good prediction [Bifet and Gavalda, 2009]. This computationally efficient and memory efficient tree architecture is therefore an asset in dataset shift context where models or parts of the models have to be retrained regularly.

Chapter 4

Dataset exploration

This work has been the opportunity to study a real world credit card transactions dataset containing all the transactions issued by belgian credit cards between March and May 2015. Having access to this proprietary dataset is an asset in several ways: the evaluations done mirror the difficulties of credit card fraud detection observed by experts, the fraudulent patterns of the dataset reflect the fraudsters strategies identified, etc.

We introduced the challenges of credit card fraud detection, from both practice and research point of views, and the solutions of the litterature to tackle these challenges in respectively chapters 1, 2 and 3. The following chapter will be the occasion to describe this belgian credit card transaction dataset:

- First, we study features correlations and features importances in the context of a credit card fraud detection task for a feature selection process. The goal of this feature selection process is to have faster computations without losing too much performances.
- Second, we introduce a method in order to quantify the temporal covariate shift in the dataset. Using this method, we show that the covariate shift within the face-to-face transactions strictly follows calendar-based patterns. We tried to incorporate this contextual knowledge within a credit card fraud detection task in a genuine way. This incorporation haven't lead to a significant increase of credit card fraud detection. This work has been published in [Lucas et al., 2019c].

4.1 Important features and correlated features

4.1.1 Transactional features

Features description

Throughout this work, we explore a dataset containing all the transactions issued by belgian credit card during between March and May 2015. The credit card transactions contain 29 features. These features describe the card-holder, the merchant and the transaction that connects them in the bipartite graph of transactions. For confidentiality reasons, we can not list all the transactions attributes of the proprietary dataset, but we list the most important ones for this thesis.

The card-holder attributes are describing the card-holder or the card:

newPanID: A categorical ID identifying the card-holder. There are 3.5 millions different card-holder IDs.

AGE: This numerical feature refers to the age of the card-holder.

GENDER: This categorical feature refers to the gender of the card-holder.

CARD-EXPIRY: This ordinal feature is refering to the month of expiration of the credit card.

PROVINCE-CODE, DISTRICT-CODE, INS-CODE, ZIP, CITY: These categorical features refer to the residence of the card-holder with various degree of precision.

$$CITY \subset ZIP \subset INSCODE \subset DISTRICTCODE \subset PROVINCECODE$$

COUNTRY: This categorical feature refers to the country of the card-holder. Unsurprisingly, more than 95% of the transactions are made by card-holders who claim to be belgian.

The merchant attributes are:

TERM-MIDUID: A categorical ID identifying the merchant. There are 300 000 merchants ID in the dataset.

TERM-MCC: This categorical feature refers to the merchant category code.

TERM-COUNTRY: This categorical feature refers to the country of the merchant. We observe that 60% of the fraudulent transactions are done with terminals from USA or great britain.

Finally, the transactions attributes are:

TX-AMOUNT: This numerical feature refers to the amount of the transaction. We observed that fraudulent transactions amounts have higher standard deviation than genuine transactions amounts. However genuine and fraudulent transactions have a similar average amount (around 40€)

TX-DATETIME: This feature refers to the time of the transaction. Additional feature can be extracted from the time of the transaction like the hour of the day, the day of the week or the time elapsed between two consecutive transactions (referred as 'time-delta' in this work)

TX-ECOM: This binary feature tells if the transaction is a face-to-face or e-commerce transaction.

TX-3D-SECURE: This binary feature tells if the e-commerce transaction triggered a 3D-SECURE authentication process. It is equal to FALSE for all face-to-face transactions

TX-EMV: This binary feature tells if the face-to-face transaction triggered a PIN authentication. It is equal to FALSE for all e-commerce transactions.

CARD-AUTHENTICATION: This categorical feature refers to the authentication process triggered for the transaction.

TX-CARD-ENTRY-MODE: This categorical feature describes the interaction between the credit card and the terminal in a face-to-face transaction.

TX-FRAUD: This binary feature tells if the transaction is a fraudulent transaction. There are 4 fraudulent transactions for 10.000 face-to-face transactions and 3 fraudulent transactions for 1000 e-commerce transactions (see table 2.1).

Features correlation

Among the 29 features of the dataset, we can imagine that some features may contain the dupicate informations. For example, the 5 features describing the residence of the card-holder should be highly correlated.

We plotted the Pearson's correlation for each pair of features (see figures 4.1 and 4.2). The Pearson's correlation between two features x and y is equal to:

$$\rho_{x,y} = \frac{cov(x,y)}{\sigma_x \sigma_y}$$

with σ_x and σ_y the standard deviations of x and y and $cov(x, y)$ the covariance of x and y[1]. It takes values between -1 and 1 where 1 is total positive linear correlation, -1 is total negative linear correlation and 0 is no linear correlation.

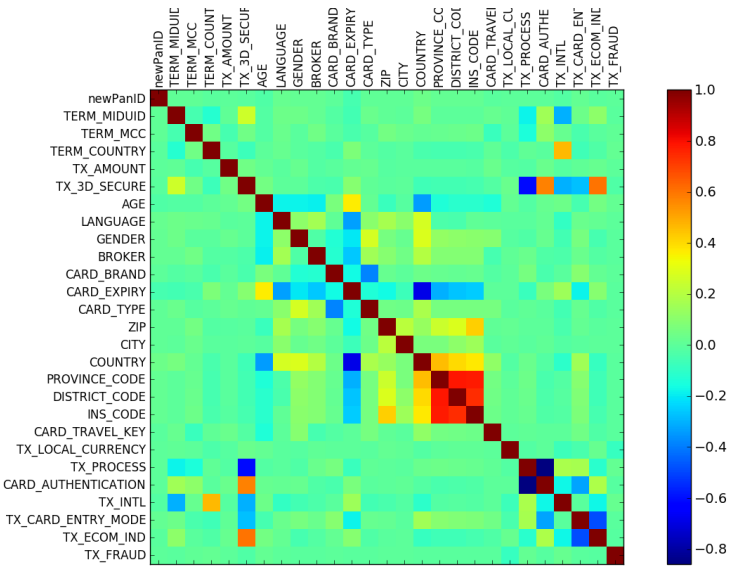


Figure 4.1: Correlation matrix for the e-commerce transactions

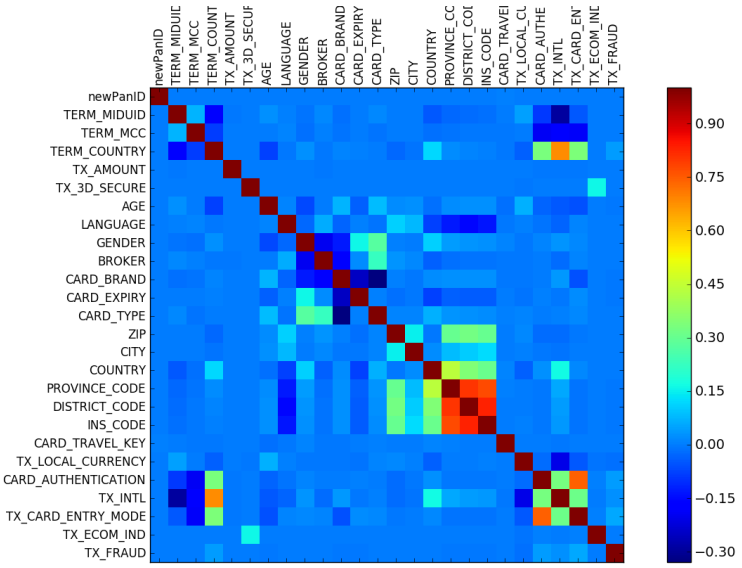


Figure 4.2: Correlation matrix for the face-to-face transactions

[1] $cov(x, y) = \mathbb{E}[(x - \mathbb{E}(x)) * (y - \mathbb{E}(y))]$

We can observe in the correlation matrices (see figures 4.1 and 4.2) that:

- Indeed, features related to the residence of the card-holder are correlated with each other.
- CARD-ENTRY-MODE and TX-ECOM-IND are highly correlated with each other. Indeed, these features describe respectively the face-to-face and the e-commerce transactions and contain a default value for respectively e-commerce and face-to-face transactions.
- TERM-COUNTRY and TX-INTL are unsurprisingly correlated: since most of the card-holders live in belgium, when TERM-COUNTRY isn't belgium, then TX-INTL has good chance to be equal to TRUE.
- CARD-AUTHENTICATION, CARD-ENTRY-MODE, TX-ECOM-IND, TX-EMV and TX-3D-SECURE all describe the authentication process used by the card-holder either in e-commerce or face-to-face context and are therefore correlated with each other.

We observed in this correlation study that some group of features contain similar informations. These correlations may be leveraged in a feature selection process.

4.1.2 Feature importances and classifiers comparison

Feature importances for tree based classifiers

Information about the importance of each features for the classification can be obtained from classifiers. These feature importances can be calculated in several ways:

Mean decrease in accuracy: Also called permutation importance, this measure is classifier agnostic. It consists in randomly permutting the values of a feature and measuring the decrease in accuracy before and after random permutation of the feature values.

Mean decrease in impurity: Also called Gini importance, this measure is specific to tree-based classifiers. For a given node, it is defined as the total decrease in Gini impurity weighted by the probability of reaching that node, which is approximated by the proportion of samples reaching that node averaged over all trees of the ensemble. At each split in each tree, the improvement in the split-criterion is attributed to the splitting variable and is accumulated over all the split of all the trees in the forest. The Gini impurity is the probability of incorrectly classifying an element of the dataset if

it were randomly labeled according to the class distribution in the dataset. It is calculated as:

$$Gini = \sum_{i=1}^{n} classp(i) * (1 - p(i))$$

In this work we used the Gini importance as it is the one calculated in the scikit-learn library. We calculated the feature importance of a random forest classifier for the task of detecting fraudulent e-commerce and face-to-face transactions (see figures 4.3 and 4.4). The hyperparameters of the random forests are the default ones: 5 features per tree, 100 trees and no tree pruning.

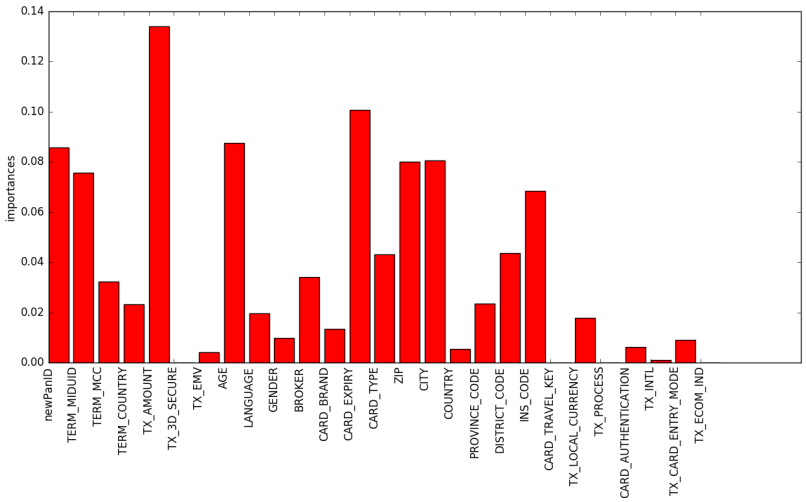


Figure 4.3: Feature importance of a random forest classifier for face-to-face transactions

- The first observation to make is that the amount of the transaction is a key feature for fraud detection. It is the most important feature for both face-to-face and e-commerce fraud detection.
- The second most important group of features is the card-holder identifiers such as newPanID, AGE, residence based features (CITY, INSCODE,...) and CARD-EXPIRY. This observation is valid for e-commerce and face-to-face transactions.
- The features describing the transaction properties (TX-LOCAL-CURRENCY, TX-EMV, CARD-TRAVEL-KEY,..) are shown to have smaller importance.

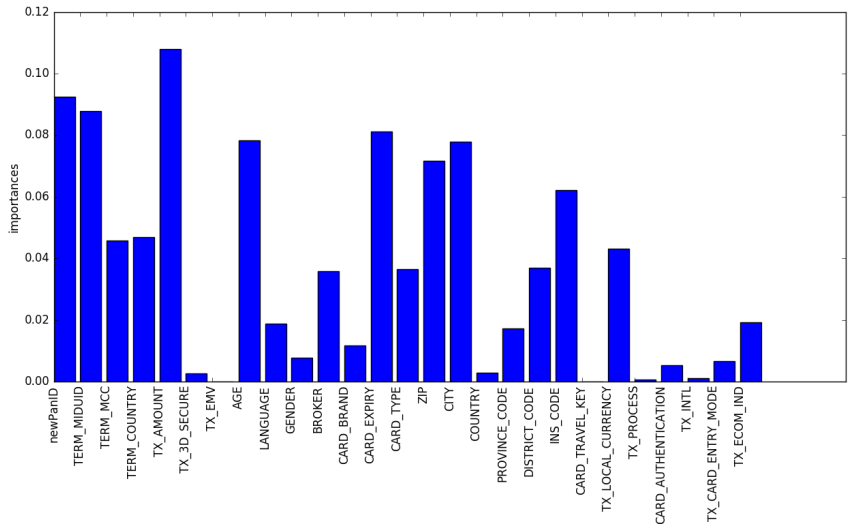


Figure 4.4: Feature importance of a random forest classifier for e-commerce transactions

Overall, there is a big discrepancy in feature importance among the features of the belgian credit card transactions dataset. In the next section, we will try to leverage this discrepancy for feature selection.

Classifiers & feature sets comparison

The transactions of the belgian credit card transactions dataset contain a variety of attributes. A big number of features slows down classifiers: The complexity of random forest or gradient boosting training for example is linear with the number of features.

Moreover, we observed in the previous sections that some features are correlated with each other: the information of these features is partly reproduced and contained in several features.

We assumed that a smaller feature set would increase the training speed without hindering too much the classifiation efficiency. In order to test that, we constructed a smaller feature set without weaker features or features containing duplicate information.

The feature set containing all the feature is referred to as the "raw" feature set. Additionnaly, we created a "refined" smaller feature set. The features contained in the "refined" feature set are presented in table 4.1.

We compared the computation times and the detection efficiencies with the

Card holder	Transaction	Terminal
age, gender, broker, card-expiry, city	authentication, card entry mode, time, amount, e-commerce type, PIN check	merchant country, merchant category

Table 4.1: Refined feature set

"raw" feature set and with the "refined" feature set for e-commerce and face-to-face fraud detection using two classifiers: Adaboost and random forest. The experiments were done using the aforementioned Spring 2015 belgian credit card transactions dataset.

The hyperparameters ranges for Adaboost and random forest grid searches are presented in table 4.2 and 4.3.

n-trees	n-features	min-samples-leaf	max depth
$\{300\}$	$\{1, 7, 13\}$	$\{1, 20, 40\}$	$\{4, None\}$

Table 4.2: Random forest grid search

tree numbers	learning rate	tolerance for stopping	max tree depth
$\{100, 400\}$	$\{0.1, 1, 100\}$	$\{10, 100\}$	$\{1, 4\}$

Table 4.3: Adaboost grid search

The training, testing and evaluation set were split temporally as shown in table 4.4. The reasoning behind the gap of one week was that in the real world fraud detection systems, human investigators have to manually verify the alerts generated by the classifiers. Since this process takes time, the ground truth is delayed by about one week.

	Start Date	End Date
Training Set	01.03.2015	26.04.2015
Validation Set	27.04.2015	30.04.2015
Week gap	01.05.2015	07.05.2015
Testing Set	08.05.2015	31.05.2015

Table 4.4: Dataset splitting

We observe on table 4.5 that refining the feature set allows for a faster training time at the cost of prediction efficiency. More precisely, the classifier efficiency measured with the ROC-AUC decreases to a maximum of 9% but this diminution of detection is accompanied by a diminution of computation time of up to 55%.

		e-commerce		face-to-face	
		ROC-AUC	computation time	ROC-AUC	computation time
random forest	raw	0.79	2367s	0.85	2530s
	refined	0.72	1739s	0.79	1811s
Adaboost	raw	0.80	11660s	0.88	13228s
	refined	0.74	5178s	0.84	6027s

Table 4.5: Comparison of prediction efficiency for 'raw' and 'refined' feature sets

Throughout this work, for faster computation time, we used the refined feature set as base feature set.

4.1.3 Comparison of sampling strategy to reduce class imbalance

Evaluation of sampling methods for our application

We tried differents sampling methods and parameters in order to determine which was most adapted for our credit card fraud detection task. For practical reasons since it increases the learning time and the risk of overfitting, we chose to not try oversampling.

We compared the value of pk100 for different strategies or parameters for the detection of fraudulent transactions in 15 consecutive testing days using a random forest algorithm. Pk100 (precision at 100) is a business metric used in production that counts the number of fraud in the first 100 alerts raised by the learning algorithm.

We started by trying different minority/majority class ratios for undersampling. The reasoning was that with the particularly big class imbalance present in our application, rebalancing the classes to 50/50 with undersampling would delete too much examples from the majority class and therefore decrease significantly the amount of information in the remaining majority class examples.

Throughout this section, the classifiers used in order to evaluate the repercussions of the sampling methods are random forest classifiers with the following parameters:

- n-trees=300
- n-features per tree=7
- depth = maximum depth

We observed in figure 4.5 that when we decreased too much the class imbalance the average performance over days of the classifier decreased: We assumed that the number of majority class examples became too small to provide enough information for the learning algorithm to generalize the majority class distribution.

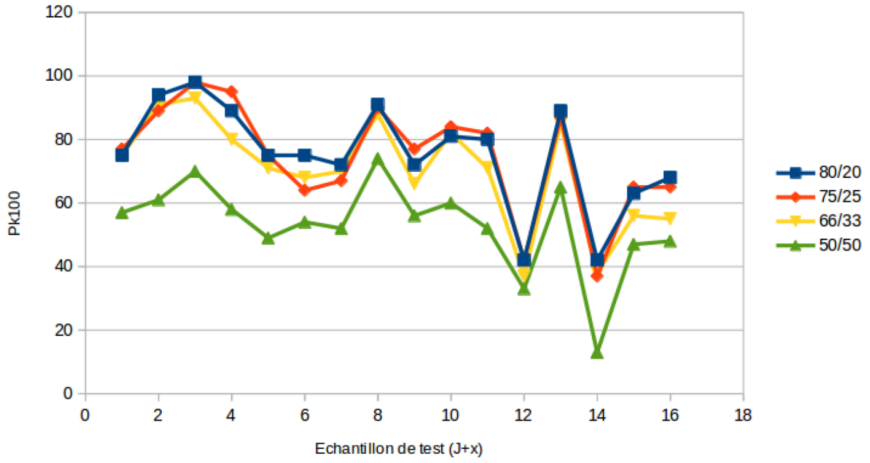


Figure 4.5: Fraud detection performance over days for different degree of undersampling. *(The numbers 80/20, 75/25, 66/33 and 50/50 refers to the class ratio after undersampling.)*

Afterwards, we tried the SMOTE strategy ([Chawla et al., 2002], highlighted in figure 3.1) and observed in figure 4.6 that the predictions with the SMOTE strategy (*in blue*) were comparable, if not below, the predictions with a basic 80/20 undersampling strategy (*in red*). SMOTE is an elegant strategy in order to deal with an unbalanced dataset, although for our application it seemed to not be the best solution and to not be worth the effort.

Finally, in most of the experiments presented in this PhD thesis, the rebalancing technique chosen is undersampling with a final ratio of 90/10 for the majority/minority effectives. We wanted to decrease the class imbalance in the dataset without taking away too many majority class examples.

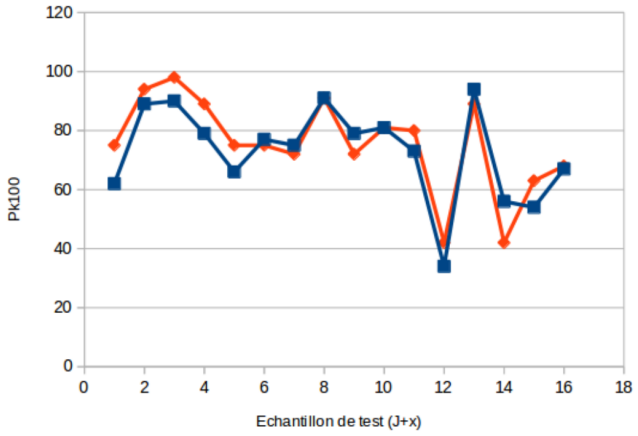


Figure 4.6: Fraud detection performance over days for SMOTE *(in blue)* and undersampling 80/20 *(in red)*.

4.1.4 Sequential dataset

Transaction sequences

Credit card transactions datasets are sequentials datasets. The credit card transactions can be grouped by the card used for the transaction or by the seller of a particular transaction. Therefore the sequence of transactions from card-holders or terminals can be studied with the hope to find patterns characterizing a genuine or a fraudulent behaviour.

We could observe on the feature importance graph (see figures 4.4 and 4.3) that the features identifying the seller (TERM-MIDUID) are very relevant for credit card fraud detection. Modelling the sequences of transactions from the buyer and the seller is an asset for credit card fraud detection.

Moreover, the timing of the transactions is very important for detecting fraudulent transactions. Unsurprisingly, we observe on figure 4.7 [2] that the face-to-face transactions occuring between 11 p.m. and 6 a.m. account for around 5% of the transactions (the shop are supposed to be closed) and for more than 40% of the fraudulent transactions.

[2]black bars on the figure (scale on the left) refer to the percentage of transactions in this category, blue bars on the figures (scale on the left too) refer to the percentage of fraudulent transactions in this category, red bars on the figures (scale on the right) refers to the ratio of fraudulent transactions for each category, each hour in this figure.

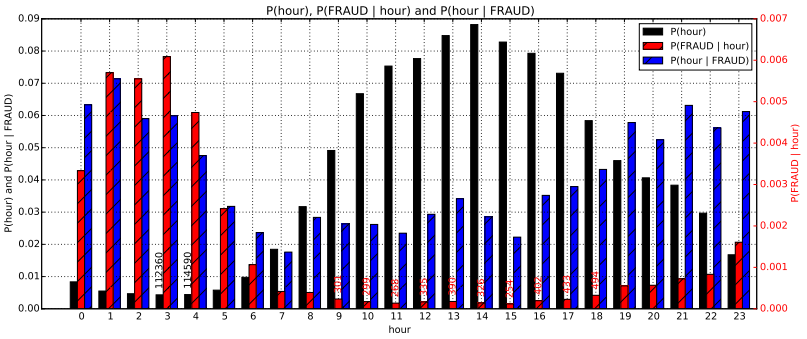


Figure 4.7: Categorical correlation between the time feature and the class of face-to-face transactions

Sequential patterns

By studying the lengths of the sequences in the dataset we could show that the 3.5 millions card-holder sequences average 13.5 transactions per sequence with significant variations: the longest sequences almost reach 10^4 transactions. Among these card holder sequences, 12853 contains the 84502 fraudulent transactions of the dataset.

Figure 4.8 shows an histogram of the time elapsed between the first and the last fraudulent transactions of the card holder sequences that contain at least a fraud. We observe on the most left bar that around 3200 card holders have a fraudulent sequence duration of 0s: these are card holder with only 1 fraudulent transaction in their sequences of transactions, the first fraudulent transaction is also the last. By cumulating bar effectives, we also observe that more than 90% of the fraudulent sequences last 28 hours or less (100 000s). The majority of the fraudulent sequences contain fraudulent transactions appearing within one day.

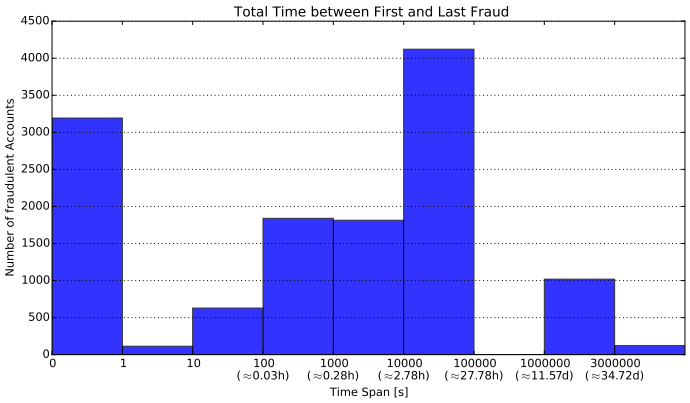


Figure 4.8: Time elapsed between first and last fraudulent transaction for each fraudulent accounts

The sequential aspect of credit card transactions dataset is crucial from both buyers and sellers perspectives for efficient credit card fraud detection. Seeing the credit card transaction dataset as a sequential dataset and identifiying previously observed patterns is a strong asset for credit card fraud detection.

4.2 Covariate shift quantification for credit card fraud detection

We proposed a covariate shift (change in $p(x, y)$) detection protocol that we used in order to characterize the covariate shift in our credit card transactions dataset.

Briefly we aim to build a distance matrix between subparts of the dataset (days in our case) by trying to predict them against each other. For example, in order to raise a similarity value for a pair of days of our dataset, we evaluate the performance (with Matthews correlation coefficient) of a random forest classifier for detecting from which of the days the testing transaction comes from. If the performance of the random forest classifier is good that means that the transactions from the two days are easy to differenciate, these two days present different buying behaviours. One the other hand if the classifier is bad, that means that the two days present similar transactions. The distance matrix obtain after having compared each pair of day can then be used for clustering the part of the dataset that present a similar buying behaviour.

In this section, we present a method to quantify the day-by-day covariate shift within the face-to-face credit card transactions dataset. In practice, we aim

to differenciate the days of the dataset by trying to guess the origin day of each transaction and measuring the efficiency of the classification. The more efficient the classification, the more different the buying behaviour between two days, and vice versa. Therefore, we obtained a distance matrix characterizing the covariate shift.

4.2.1 Building a distance matrix between days

We aimed to use a distance matrix in order to quantify the possible covariate shift happening over the 92 days of the dataset.

For this purpose we built and evaluated learners that try to tell from which day of the considered pair of days each transaction belongs. If the learner achieved to detect the day of the testing set transactions, it means that the transactions from each of the two days considered were easy to differentiate. We assumed that these two days are following two different probability distributions: there was a covariate shift between these two days. On the other hand, if the two days were hard to differenciate, this means that their transactions are similar: there was no covariate shift between these days.

For each pair of days, a random forest classifier was trained using 20000 transactions from each day (40000 in total) in order to detect the day of each transaction. Afterwards, this random forest classifier was asked to classify a test set containing 5000 transactions from each day (10000 in total). We chose to import the same number of transactions from each days even if some days contain more transactions than others (approximately between 300 000 and 800 000 per day in total) instead of importing a number of transactions proportional to the number of transactions within the considered day. Therefore the model prediction isn't affected by the class effectives but only by the buying behaviours within each day.

The classification was evaluated using the Matthews correlation coefficient (MCC). As mentioned in section 2.2, the Matthews correlation coefficient is a confusion matrix based metric that can take values between -1 and 1. A value of MCC close to 1 means that the classifier is good at differenciating the pair of days considered, the days are easy to differenciate since there is a covariate shift between them. A value of MCC close to 0 means that the classifier isn't able to differenciate the transactions of the pair of days considered. In this particular classification task, it would not make sense that the classifier have a value of MCC close to -1. It would mean that the classifier is predicting the opposite between training and testing set.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

In the end, this MCC value was used in order to train and evaluate $92 * 91/2$ classifiers in order to build a 92x92 distance matrix characterizing the covariate shift between the 92 days of the dataset. Figure 4.9 shows a zoomed view of the distance matrix centered on the covariate shift within the days from 01.03.2015 to 03.04.2015. The full view of the distance matrix on the whole dataset timespan is included on the next page.

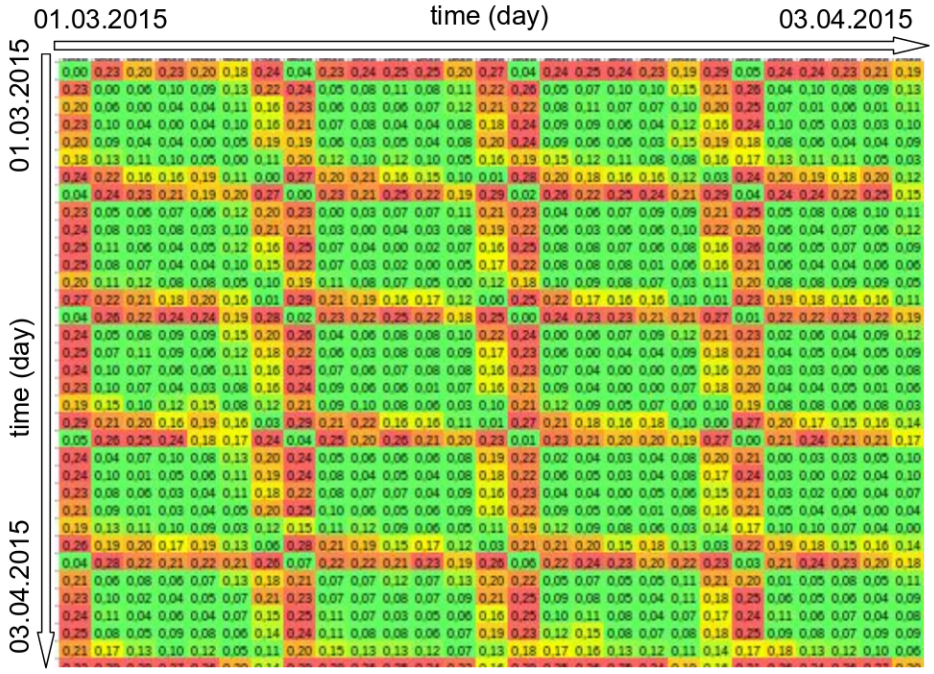
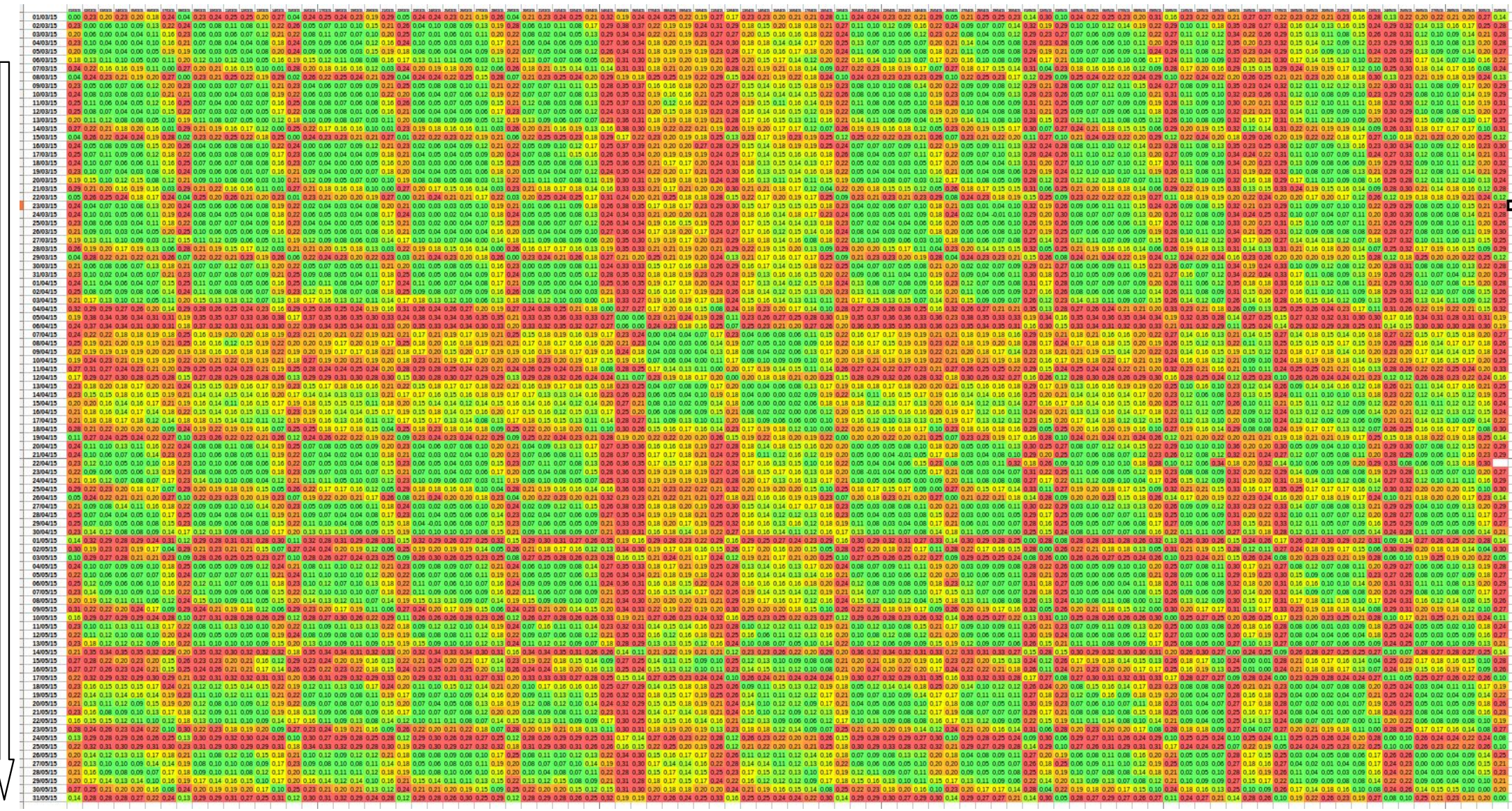


Figure 4.9: Zoomed distance matrix (covariate shift) for the face-to-face transactions. *(centile based coloration: green ⇔ similar days (MCC≈0), red ⇔ dataset shift (MCC≈1)).*


01.03.2015
time (day)
31.05.2015
01.03.2015
time (day)
31.05.2015

4.2.2 Agglomerative clustering using the distance matrix between days

Regularities can be observed in the covariate shift (see figure 4.9 in appendix). With genuine visual identification, 4 clusters of days that are similar to each other and dissimilar to other days can be spotted. We could correlate these clusters with calendar event (Dataset of belgian transactions):

- **Working days:** During these days, people are working and merchants are mostly open.
- **Saturdays:** During these days, people are not working and merchants are mostly open.
- **Sundays:** During these days, people are not working and merchants are mostly closed. We observe that some religious holidays (catholic) seem to share the same dataset shift (1st may: labor Day, 14th may: Ascent and 25th may: Pentecost).
- **School holidays:** During these days, people are working and the merchants are mostly open but the children are not at school. Some parents take holidays to look after their children or travel with them.

This clustering is based on a qualitative observation of the distance matrix between days (figure 4.9 in appendix) and may be biased: the patterns visually identified may have been inferred because we had preconceptions about the type of days in the dataset. Therefore, we performed a hierarchical clustering on the distance matrix.

The dendrogram created with agglomerative clustering can be cut at different levels. These levels correspond to different number of clusters. In figure 4.10 we observe that the drop of the average intercluster distance sharply decreases after 5 clusters. This information would guide the decision towards a 5 clusters day modeling. However, since 4 cluster isn't completely irrelevant (big intercluster distance decrease between 3 and 4 cluster), and since we observed with the previous qualitative clustering that 4 clusters match well the calendar particularities, we chose to differentiate the days in 4 clusters using hierarchical clustering (see figure 4.6).

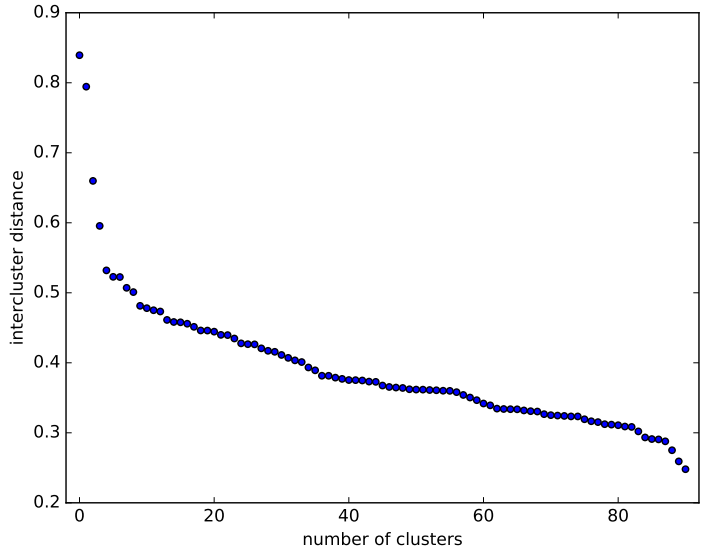


Figure 4.10: Diminution of the average inter-cluster distance with the increase of the number of clusters

The output of the hierarchical clustering is shown in table 4.6. The 4 clusters matched almost perfectly the calendar-based clusters previously identified.

0: Week days (working days)

1: Sundays and work holidays (Ascent, monday Pentecost and Labor day): shops are closed

2: Saturdays and most of the Fridays : days of the end of the week where shops are open

3: Week days of eastern school holidays

- Surprisingly, 9 Fridays out of the 12 non holiday Fridays are clustered with Saturdays (cluster 2). We expected initially the Fridays to be clustered with the rest of the working days (cluster 0)
- Among the 42 days identified as working days of the middle of the week (cluster 0), 95% follow the pre-identified pattern: 2 of them are Fridays which are supposed to be in the cluster 2. Moreover, 2 working days adjacent to holidays where shops are closed (cluster 1) are clustered with Fridays and Saturdays (2) instead of working days (0)

[3] 1st may: Labor day
[4] 14th may: Ascent
[5] 25th may: Pentecost

Week	MON	TUE	WED	THU	FRI	SAT	SUN
9							1
10	0	0	0	0	0	2	1
11	0	0	0	0	2	2	1
12	0	0	0	0	2	2	1
13	0	0	0	0	2	2	1
14	0	0	0	0	2	2	1
15	1	3	3	3	3	2	1
16	3	3	3	3	2	2	1
17	0	0	0	0	0	2	1
18	0	0	0	2	1 [3]	2	1
19	0	0	0	0	2	2	1
20	0	0	2	1[4]	2	2	1
21	0	0	0	0	2	2	1
22	1[5]	0	0	0	2	2	1

Table 4.6: Agglomerative clustering of the days using the distance matrix obtained by classifying each day against every other day

- Among the 18 days identified as holidays (cluster 1), 17 are effectively holidays. Unexpectedly, the first Monday of spring holidays (week 15) is clustered with holidays.
- The days of the eastern school holidays pre-identified pattern are successfully clustered together by the hierarchical clustering algorithm in the cluster 3.

In the end we observed that clustering the days of the dataset using the covariate shift distance matrix, be it visually or with hierarchical clustering, outputs similar results. Indeed, the clustering patterns discovered match almost perfectly the event identified in the belgian calendar.

4.2.3 Incorporation of dataset shift knowledge

In order to leverage the knowledge of the covariate shift between days, we incorporate it as a new categorical feature with one modality for each identified cluster.

We repeated 5 times an experimental protocol where the face-to-face transactions from 7 consecutive days were used as a train set and the face-to-face transactions from 7 consecutive days with a gap of 7 days after the train set were used as a test set. We evaluated the addition of the dataset shift feature by computing the Precision-Recall AUC and the ROC AUC for random forest

	PR AUC		ROC AUC	
test set	without	with	without	with
16/03-22/03 (week 12)	0.190	0.175	0.964	0.961
23/03-29/03 (week 13)	0.288	0.303	0.969	0.973
30/03-05/04 (week 14)	0.131	0.142	0.939	0.945
06/04-12/04 (week 15)	0.256	0.273	0.914	0.927
13/04-19/04 (week 16)	0.131	0.126	0.946	0.940
average	0.199	0.204	0.946	0.949

Table 4.7: AUC variations with the addition of the covariate shift feature for different testing periods

classifiers with and without the dataset shift feature.

We chose to separate the time period corresponding to the training and validation set and the time period corresponding to the testing set with a gap of 7 days.

The random forest hyperparameters are described in table 4.8. The same hyperparameters were used for the construction of the distance matrix between days.

n-trees	n-features	min-samples-leaf
100	*sqrt*	10

Table 4.8: Random forest hyperparameters

We observed in table 4.7 that adding the covariate shift information as a new feature for the classification increased the precision-recall AUC by 2.5%. The ROC AUC was also slightly increased. However the increase observed is weak and not consistent over weeks. Besides, we don't seem to observe any correlation between the calendar events and the increase/decrease observed: It is not clear that for example, testing set belonging to the eastern holidays (week 15 and 16) are less sensitive to the addition of covariate shift information as a new feature.

Moreover, this improvement may partly have been a consequence of the weak feature set used and may have disappeared when using stronger feature engineering strategies such as feature aggregation [Bahnsen et al., 2016][Whitrow et al., 2008] or sequence modelling with hidden Markov Models [Lucas et al., 2019a] or recurrent neural networks [Jurgovsky et al., 2018].

4.2.4 Conclusion

In this section we proposed a strategy to quantify the covariate shift in a temporal dataset. This strategy consists in classifying the transactions of each day against every other days: if the classification is efficient then the days are different and there is a covariate shift between them. On the other hand, if the classification is not efficient, the days are similar. This strategy allows us to build a distance matrix characterizing the covariate shift between days.

Afterwards, we used an agglomerative clustering algorithm on the distance matrix between days. We showed that in the case of the belgian face-to-face credit card transactions the dataset shift matches almost exactly the calendar events. We identify 4 types of days in the credit card transactions dataset: 'working days', 'saturdays', 'sundays' and 'school holiday'.

Using a random forest classifier, we showed that integrating the information of the type of the day previously identified increases the Precision-Recall AUC by a small percentage (2.5%).

4.3 Summary

In this chapter, we started by studying the attributes of credit cards transactions with leverages such as covariance matrix and random forest feature importance vectors. Using these studies, we could create a refined feature set that would lead to faster computation time and memory efficiency at the cost of a bit of fraud detection efficiency. The refined feature set is shown in table 4.1.

We also tried different solutions for reducing class imbalance: SMOTE and undersampling along with different values of target class ratio. We chose to avoid oversampling because it increases computation time along with overfitting risks. We settled on undersampling the majority class up to a target class ratio of 90:10 after having observed that decreasing too much the class imbalance leads to the deletion of too many examples in the dataset.

We finished this chapter by presenting a study of the covariate shift within the face-to-face transactions. We could identify covariate shift patterns matching calendar events. We tried to integrate this covariate shift knowledge in the credit card fraud detection process and observed a slight increase in precision-recall AUC.

Chapter 5

Multiple perspectives feature engineering

Throughout the previous chapters, we stated the importance of the feature engineering leverage for credit card fraud detection. Many feature engineering strategies have been imagined in the literature as described in section 3.2. The transaction aggregation strategy introduced by [Whitrow et al., 2008] is the most adopted approach in the field.

Moreover, we became aware that sequential patterns are ubiquitous in credit card fraud detection. Genuine behaviours are ruled by temporal patterns (see section 4.2 and fraudulent strategies are identified through sequential patterns (see section 4.1.4).

In this chapter, we want to leverage the sequential patterns discovered through the creation of features. Therefore, we present an approach to model a sequence of credit card transactions from three different perspectives, namely:

(i) The sequence contains or doesn't contain a fraudulent transaction.

(ii) The sequence is obtained by fixing the card-holder or the payment terminal.

(iii) It is a sequence of spent amount or of elapsed time between the current and previous transactions.

Combinations of the three binary perspectives give eight sets of sequences from the (training) set of transactions. Each one of these sequences is modelled with a hidden Markov model (HMM). Each HMM associates a likelihood to a transaction given its sequence of previous transactions. These likelihoods are used as additional features in a random forest classifier for fraud detection.

This multiple perspectives HMM-based approach enables an automatic feature engineering in order to model the sequential properties of the dataset with respect to the classification task. This approach allows for a 9.3% increase in the precision-recall AUC on the e-commerce transactions and a 18.1% increase on the face-to-face transactions compared to the state of the art feature engineering strategy for credit card fraud detection.

5.1 Baseline: transaction aggregation strategy

As stated in section 3, one of the difficulty of credit card fraud detection is the fact that the feature set describing a credit card transaction usually ignores detailed sequential information. Typical models only use raw transactional features, such as time, amount, merchant category, etc.

Consequently, [Whitrow et al., 2008] create descriptive statistics as features in order to include historical knowledge. These descriptive features can be for example the number of transactions or the total amount spent by the card-holder in the past 24 hours for a given merchant category or country. [Bahnsen et al., 2016] considered [Whitrow et al., 2008] strategy to epitomized the state of the art feature engineering technique for credit card fraud detection.

5.1.1 Construction of features based on transaction aggregation strategy

The state of the art feature engineering technique for credit card fraud detection creates descriptive features using the history of the card-holder [Whitrow et al., 2008], [Bahnsen et al., 2016].

In order to enrich a transaction with statistics constructed with transaction aggregation strategy, 3 objects are needed:

A sequence of previous transactions: It can be the sequence of previous transactions from the buyer or the seller. The literature contains mostly feature based on the buyers sequences, however we could show that the seller perspective is very relevant for credit card fraud detection. This sequence of transaction will be filtered and used for the calculation of statistics.

A filter: This filter aims to consider only the relevant past transactions for the construction of the descriptive statistic. It usually contains a time limit, for example only considering past transactions from the last 24 hours, from the last week or the last month. It may also contain a combination of one or several categorical features: For example, in order for the transactions

to be considered, they must have been done in the same country and/or with the same type of merchant and/or same type of authentication, etc.

An operation: In order to extract information from the filtered sequence of transactions, an aggregation operation has to be performed. This operation can use information from the features of the transactions. It can be for example the number of transactions in the sequence, the total amount spent in the sequence, the standard deviation of a chosen feature in the sequence, etc.

5.1.2 Transaction aggregation strategy improves fraud detection

Naive use of transaction aggregation strategy (aggCH)

We implemented the transactions aggregation strategy in order to verify the increase in detection claimed in [Whitrow et al., 2008] and [Bahnsen et al., 2016].

In addition to the 13 refined features of the credit card transaction dataset (see section 4.1.2 for the construction of the refined feature set), we calculated the value of 4 features based on the transaction aggregation strategy applied to the card-holder. These 4 features based on the card-holder history will be referred to as the state of the art feature engineering in later studies.

Card holder	Transaction	Terminal
age, gender, broker, card-expiry, city	authentication, card entry mode, time, amount, e-commerce type, PIN check	merchant country, merchant category

Table 5.1: Refined feature set

Feature	Signification
AGGCH1	# transactions issued by user in 24h.
AGGCH2	Amount spent by user in 24h.
AGGCH3	# transactions in the country in 24h.
AGGCH4	Amount spent in the country in 24h.

Table 5.2: Aggregated features centered on the card-holders

We tested the addition of these features for the face-to-face transactions and for the e-commerce transactions. We measured the precision-recall AUCs and ROC AUCs before and after inclusion of those aggregated features. We call

aggCH the set of features based on transaction aggregation strategy on the cardholder sequences.

We considered 3 types of classifier to study the increase in detection when using transaction aggregation strategy: random forest classifier, logistic regression classifier and Adaboost classifier. We reproduced each prediction 3 times in order to monitor the variation of AUCs among the runs. The hyperparameters ranges chosen for the grid search optimization of the classifiers is presented in tables 5.3, 5.4 and 5.5.

n-trees	n-features	min-samples-leaf	max depth
$\{300\}$	$\{1, 7, 13\}$	$\{1, 20, 40\}$	$\{4, None\}$

Table 5.3: Random forest grid search

C parameter	penalty	tolerance for stopping
$\{1, 10, 100\}$	$\{l1, l2\}$	$\{10, 100\}$

Table 5.4: Logistic regression grid search

tree numbers	learning rate	tolerance for stopping	max tree depth
$\{100, 400\}$	$\{0.1, 1, 100\}$	$\{10, 100\}$	$\{1, 4\}$

Table 5.5: Adaboost grid search

The training, testing and evaluation set were split temporally as shown in table 4.4.

The consequences of the incorporation of transaction aggregation strategy based features on classifiers prediction were evaluated using the Precision-Recall AUC and ROC AUC metrics described in section 2.2.

Experiments

	precision-recall AUC		ROC AUC	
	without aggCH	with aggCH	without aggCH	with aggCH
e-commerce	0.212 ± 0.0009	0.343 ± 0.001	0.935 ± 0.00003	0.950 ± 0.0001
face-to-face	0.082 ± 0.001	0.144 ± 0.001	0.950 ± 0.0005	0.964 ± 0.00005

Table 5.6: Addition of *"aggCH"* features to a random forest classifier

	precision-recall AUC		ROC AUC	
	without aggCH	with aggCH	without aggCH	with aggCH
e-commerce	0.015 ± 0.0001	0.092 ± 0.005	0.812 ± 0.002	0.856 ± 0.014
face-to-face	0.0094 ± 0.0001	0.014 ± 0.001	0.916 ± 0.002	0.923 ± 0.014

Table 5.7: Addition of *"aggCH"* features to a logistic regression classifier

	precision-recall AUC		ROC AUC	
	without aggCH	with aggCH	without aggCH	with aggCH
e-commerce	0.103 ± 0.001	0.238 ± 0.009	0.857 ± 0.003	0.873 ± 0.014
face-to-face	0.034 ± 0.003	0.048 ± 0.002	0.825 ± 0.004	0.925 ± 0.024

Table 5.8: Addition of *"aggCH"* features to an Adaboost classifier

We observe a significant improvement in prediction when incorporating features based on the transaction aggregation strategy of the card-holder sequences.

- The introduction of features based on transaction aggregation strategy causes the random forest classifier to increase its precision-recall AUC by 62% for the e-commerce transactions and 76% for the face-to-face transactions.
- The introduction of features based on transaction aggregation strategy causes the logistic regression classifier to increase its precision-recall AUC by 510% for the e-commerce transactions and 49% for the face-to-face transactions.
- The introduction of features based on transaction aggregation strategy causes the Adaboost classifier to increase its precision-recall AUC by 130% for the e-commerce transactions and 41% for the face-to-face transactions.

The small values of standard deviation ensure the validity of the conclusions.

Importance of terminal perspective

We showed in the previous section that the transaction aggregation strategy of [Whitrow et al., 2008] applied on the sequences of card-holders enables a significant improvement in credit card fraud detection.

However, the credit card transactions happen between buyers and sellers but the merchants sequences are rarely considered in the studies using transaction aggregation strategy as their state of the art.

Since modelling the terminal perspective is an important part of our work we wanted to try to build features based on transaction aggregation strategy on the sequences of the merchants and assess their relevance for credit card fraud detection. This would ensure a fair comparison with state of the art techniques when we would add the proposed HMM-based features modelling the merchants sequences to the feature set.

Feature	Signification
AGGTM1	# transactions in terminal in 24h.
AGGTM2	Amount spent in terminal in 24h.
AGGTM3	# transactions with this card type in 24h.
AGGTM4	Amount spent with this card type in 24h.

Table 5.9: Aggregated features centered on the terminals

We tested the addition of the terminal based aggregated features over the raw features **and** the card-holder based aggregated features for the face-to-face transactions and for the e-commerce transactions. We measured the precision-recall AUCs and ROC AUCs before and after inclusion of those aggregated features. We call *aggTM* the set of features based on transaction aggregation strategy on the terminal sequences.

We considered 3 types of classifier to study the increase in detection when adding the perspective of the merchants described with transaction aggregation strategy: random forest classifier, logistic regression classifier and Adaboost classifier. We reproduced each prediction 3 times in order to consider the variation of AUCs among the runs.

The models hyperparameters, evaluation metrics and dataset splitting are identical to those used in the previous section 5.1.2.

Experiments

	precision-recall AUC		ROC AUC	
	without aggTM	with aggTM	without aggTM	with aggTM
e-commerce	0.343 ± 0.001	0.383 ± 0.002	0.950 ± 0.0001	0.957 ± 0.0002
face-to-face	0.144 ± 0.001	0.167 ± 0.001	0.964 ± 0.00005	0.967 ± 0.0004

Table 5.10: Addition of *"aggTM"* features to a random forest classifier

	precision-recall AUC		ROC AUC	
	without aggTM	with aggTM	without aggTM	with aggTM
e-commerce	0.092 ± 0.005	0.096 ± 0.004	0.856 ± 0.014	0.867 ± 0.010
face-to-face	0.014 ± 0.001	0.0161 ± 0.0003	0.923 ± 0.014	0.933 ± 0.004

Table 5.11: Addition of *"aggTM"* features to a logistic regression classifier

	precision-recall AUC		ROC AUC	
	without aggTM	with aggTM	without aggTM	with aggTM
e-commerce	0.238 ± 0.009	0.263 ± 0.004	0.873 ± 0.014	0.907 ± 0.006
face-to-face	0.048 ± 0.002	0.053 ± 0.003	0.925 ± 0.024	0.83 ± 0.001

Table 5.12: Addition of *"aggTM"* features to an Adaboost classifier

- The addition of features describing the merchants sequences with transaction aggregation strategy to the raw features **plus** the card-holder based aggregated features causes the random forest classifier to increase its precision-recall AUC by 12% for the e-commerce transactions and 16% for the face-to-face transactions.
- The addition of features describing the merchants sequences with transaction aggregation strategy to the raw features **plus** the card-holder based aggregated features causes the logistic regression classifier to increase its precision-recall AUC by 4% for the e-commerce transactions and 15% for the face-to-face transactions.
- The addition of features describing the merchants sequences with transaction aggregation strategy to the raw features **plus** the card-holder based aggregated features causes the Adaboost classifier to increase its precision-recall AUC by 11% for the e-commerce transactions and 10% for the face-to-face transactions.

For each classifier except logistic regression classifier, the small relative values of standard deviation ensure the validity of the conclusions.

We observe an improvement in prediction when incorporating transaction aggregation strategy features based on terminals sequences in addition to features based on the card-holders sequences. It is encouraging since considering merchants perspective is one of the particularity of this work. Indeed, it is not considered in the majority of the credit card studies we read.

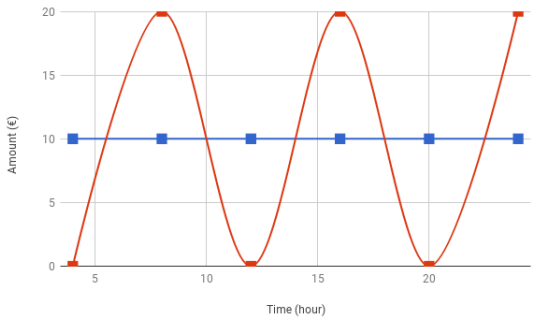


Figure 5.1: Modelling limitations of transaction aggregation strategy

5.1.3 Limitations of transaction aggregation strategy

We identified several weaknesses in the construction of features based on transaction aggregation strategy that motivated our work:

First, Aggregated features provide a descriptive statistic over a set of transactions. Such statistics do not consider fine-grained temporal dependencies between the transactions of a sequence. For example, Two card-holders may have very different buying behaviours: steady transactions amounts or oscillating transactions amounts, however the features created by the transactions aggregation strategy wont reveal these differences: the two card-holder will have the same value of mean amount and transaction frequencies (see figure 5.1). Moreover we observed common fraud patterns which start with low amount transactions for testing the card, followed by high amount transaction to empty the account: sequential information should be relevant for credit card fraud detection. Therefore we propose to use a generative graphical model for sequence modelling. The choice of hidden Markov model for our sequence modelling problem is discussed in section 5.2.1.

Second, Aggregated features are usually calculated over transactions occuring in a fixed time window (e.g. 24 h). Transaction of very different card holders do not follow such a time pattern in general. However, the number of transactions made during such a time period can vary a lot for different card-holders. Fixed size aggregated statistics can't account for that fact.

Third, These features consider only the history of the card-holder and do not exploit information of fraudulent transactions for feature engineering. However, a sequence of transactions happening at a fixed terminal can also contain valuable patterns for fraud detection.

Fourth, The choice of the descriptive feature created using the transaction aggregation strategy ([Whitrow et al., 2008], [Bahnsen et al., 2016]) is guided by expert knowledge. We would like not to be dependent to expert knowledge and we think that doing feature engineering in a supervised way (with the knowledge of the label of transactions in the training set) can be a way to create relevant features without the need of expert knowledge.

5.2 Multiple perspectives hidden Markov model based feature engineering

5.2.1 Modelling the sequences of transactions

As we have discussed before, feature aggregation strategies improve results but ignore sequence information. We want to model the sequences of transactions instead of creating descriptive features. However, we have seen in section 3.3 that several solutions to model sequences exist. This brings us to the question: Which model to choose for including sequential information?

Why hidden Markov models?

We choose to model transactions sequences using HMMs for several reasons.

- First, HMM were already used with success for credit card fraud detection in [Srivastava et al., 2008]. The authors proposed to raise anomaly scores based on sequences of amount spent by the card-holder ('big amount', 'medium amount', 'small amount'). They created an artificial credit card transaction dataset in order to show the benefits of their approach. Other authors like [Dhok, 2012] also used HMM for credit card fraud detection.

- Second, hidden Markov models are generative models, i.e. they model the joint probability $p(x, y)$. We aim to exactly model that probability and not the conditional distribution $p(y|x)$ which is estimated by discriminant classifiers like LSTM or CRF. In our case, the classification is done afterwards by a random forest, logistic regression or Adaboost classifier.

- Third, hidden Markov models are very simple models. A HMM containing 7 hidden states and a gaussian emission distribution has $49 + 14 = 63$ parameters whereas neural networks models can have thousands of parameters. This reduced number of parameters allows for a compression of the modelled behaviours. The observed behaviour is generalized into an abstracted latent behaviour. Moreover, the reduced number of parameters enables an easier interpretation of the behaviour modeled by the HMMs. For example, by looking at the probability to remain in the current hidden state,

we could observe that the genuine buying behaviours showed sequences of steady amount of transactions whereas the fraudulent buying behaviours were more volatile.

Markov property in a credit card transaction environment

As introduced in section 3.3, the term Markov property refers to the memoryless property of a stochastic process like the one described by hidden Markov models (see figure 5.2). In other words, the conditional probability distribution of future states of the process depends only upon the present state, not on the sequence of events that preceded it. Formally, it allows simplification of conditional probabilities:

$$p(t_k|t_{k-1}, t_{k-2}, .., t_1, w_{k-1}, w_{k-2}, w_1) = p(t_k|t_{k-1}) \; k \in \{1, .., n\}$$

In the case of credit card transaction sequences, the Markov assumption is questionable. On one hand some buying behaviours where the card-holder visits shops one after the other within a predefined itinerary would respect the Markov assumption. On the other hand some credit card transactions have longer term dependencies like for example the weekly supermarket visits for a family.

Moreover, other authors such as [Srivastava et al., 2008], [Dhok, 2012], [Robinson and Aria, 2018] considered the Markov assumption valid for credit card fraud detection. Therefore, we assumed that the Markov assumption holds for our credit card fraud detection task.

5.2.2 Hidden Markov models basic principles

Hidden Markov models main hypothesis is that behind the observed distribution, there is a simpler latent model that rules the sequential distribution. Therefore, sequence modelling with hidden Markov model comprises two processes. The stochastic hidden states evolution at each time step is not observed. Instead, observed variable values are generated at each time step depending on the current value of the corresponding hidden state thanks to an emission process. Hypothetically, the complexity of the observed distribution comes partly from additive Gaussian noise corrupting both the hidden states evolution process and the emission process.

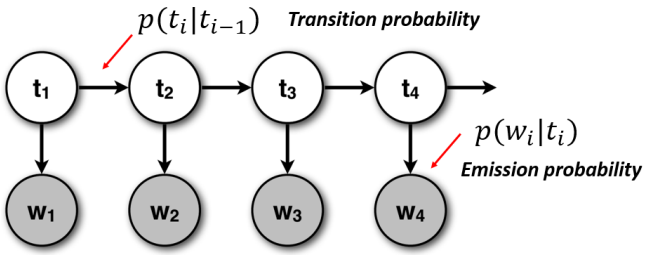


Figure 5.2: Hidden Markov model architecture (from [Lucas et al., 2019a]).

We can observe in the trellis diagram[1] (see figure 5.2) describing the HMM that they comprises two types of random variables:

- $t_i \in \{1, 2, .., m\}$ are discrete random variables : the hidden states or latent states.
- w_i are the observed variables. The conditional distribution of the observed variable with respect to the hidden states depends on the observed variable nature.

Hidden Markov models allow for a compression of the observed sequence in order to generalize the observed behaviour into an abstracted latent behaviour. It is comprised of two processes represented by matrices:

- The transition matrix $T(i, j)$ describes the transition probabilities: $T(i, j) = p(t_{k+1} = j|t_k = i),\ i, j \in \{1, .., m\}^2$. It rules the succession of the hidden states. Each row i of the transition matrix is a multinomial distribution of the next state given that the current state is i. The hidden states obey to the Markov property.
- The *emission matrix* describes the conditional distribution of the observed variables given the current hidden state. Emission probabilities distribution can be arbitrary. They can be discrete finite (multinomial distribution), or infinite (Poisson distribution), continuous (gaussian distribution, lognormal distribution) and even multidimensional. The emission matrix describes the emission probabilities: $E_i(w) = p(w_k = w|t_k = i),\ i \in \{1, ..., m\},\ w \in W$ for the discrete case[2].

The joint probability of the sequences of w_i and t_i described by the trellis diagram is:

$$p(w_1, ..., w_n, t_1, ..., t_n) = p(t_1) * p(w_1|t_1) \prod_{k=2}^{n} p(t_k|t_{k-1}) * p(w_k|t_k)$$

[1] temporal visualisation of a graphical model

[2] or $E_i(w) = p(w|t_k = i),\ i \in \{1, ..., m\},\ w \in W$ for the continuous case (w has a density distribution).

In the end, the parameters of an HMM are the transition matrix ($T(t_{k-1}, t_k)$ and the emission matrix $E_{t_k}(w_k)$ described above, plus the initial distribution of the hidden states $\pi = p(t_1 = i),\ i \in 1, ..., m$.

Using these parameters, the joint probability of w_i and t_i becomes:

$$p(w_1, .., w_n, t_1, .., t_n) = \pi * E_{t_1}(w_1) \prod_{k=2}^{n} T(t_{k-1}, t_k) * E_{t_k}(w_k)$$

Practical exemple of data modelling using an HMM A practical example of HMM is presented in figure 5.3. In this example, $t_i \in \{-1; 1\}$ and $w_i \in \mathbb{R}$. The transition matrix is:

$$\begin{pmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{pmatrix}$$

The observed variables w_i are distributed around the t_i. The emission distribution follows an uniform distribution around the :

$$w_i = t_i * (0.5 + X_i),\ X_i \to \mathcal{U}([0; 1]),\ i \in \{1, .., n\}$$

.

At each point of time $i \in \{1, .., 100\}$, an hidden state value t_i (in blue) is calculated with the transition matrix. For each hidden state t_i, a noisy observation w_i (in red) is made depending on this hidden state.

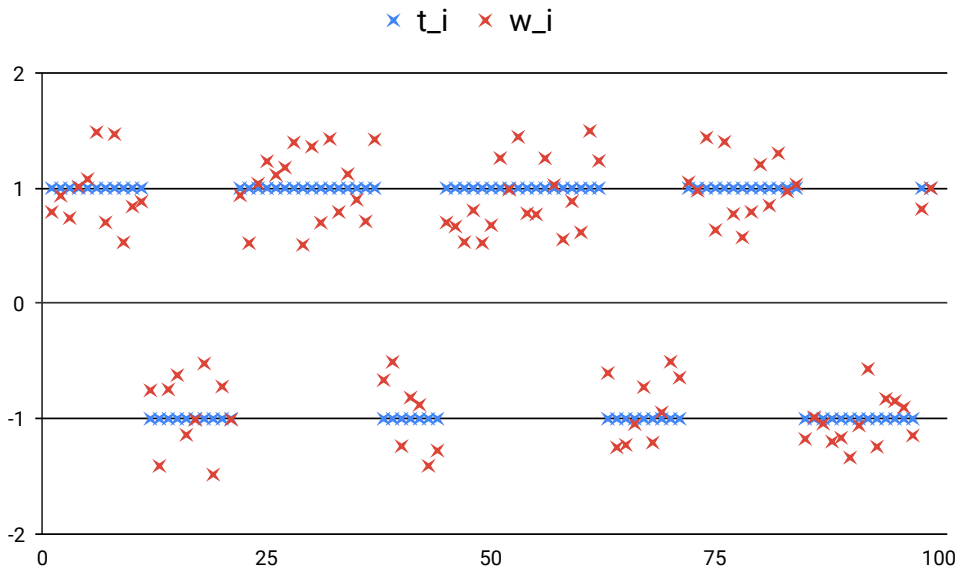


Figure 5.3: Practical example of HMM

Training hidden Markov models: Expectation Maximisation algorithm

Given a set of sequences of observed variable, the parameters of an hidden Markov model can be adjusted to reflect the distribution observed in these sequences of observed variable. The algorithm used in order to "fit" the hidden Markov model to a set of sequences is the Expectation-Maximisation algorithm.

We described earlier the hidden Markov model by $\theta = (T, E, \pi)$. Given a set of observed sequences W. The iterative Expectation-Maximisation algorithm known as the Baum-Welch algorithm finds a local maximum for $\theta^* = argmax_\theta p(W)$ (i.e. the HMM parameters θ that maximise the probability of the observed sequence).

Qualitatively, EM optimization of a model with latent (hidden) parameters consists in for randomly initialized parameters:

Expectation: Find the latent states distributions that correspond the most to the sequences of observed data. This is usually done with the help of the forward backward algorithm which leverage the Markov property in order to simplify the calculations of the conditional probabilities to observe a sequence of event given the parameters of the transition and emission matrices (also referred to as forward-backward algorithm [Viterbi, 1967]).

Maximisation: Maximise the correspondence between the latent distribution inferred during the expectation step and the parameters of the transition and emission matrices by adjusting the parameters.

The expectation and maximization steps are repeated until convergence of the graphical model to the observed data. The convergence can be monitored by observing the increase of the value of the likelihood that the set of observed sequences has been generated by the model. This likelihood increases over the iterations until it reaches a ceiling when the hyperparameters ruling the architecture of the generative model don't allow it to fit more to the set of observed sequences.

Knowing the probability of state frequencies and the probabilities to obtain the observed variables, the maximisation step is quite straightforward. However, a naive calculation of the probability for each observed sequence to have been generated by the current iteration of the HMM is very costly $O((\mathbf{m^n})$ with m the number of hidden states and n the size of each sequence). The forward-backward algorithm simplifies a lot this calculation with a recursion trick detailed below. The complexity of the expectation step becomes $O(\mathbf{n} * \mathbf{m^2})$ with the forward-backward algorithm.

The forward-backward algorithm of the expectation step assumes that the parameters of the HMM ($\theta = (T, E, pi)$) are known. The goal of the forward backward algorithm is to compute $p(t_k, w)\ \forall k \in \{1, .., n\}$ with $w = \{w_1, .., w_n\}$ a sequence of observed events. We will note $w_{i:j} = (w_i, .., w_j)$ a subsequence of events. Therefore, $w = w_{1:n}$.

The forward-backward algorithm leverages the Markov property in order to simplify the calculation of conditional probability. The simplification is done using the d-separation. Basically, two variables of a graphical model are d-separated conditioned on a third variable if they are independant conditioned on the third variable. In the case of hidden Markov model, since the distribution of hidden states follow the Markov property, two distant hidden states are independant conditioned on the knowledge of an intermediate hidden state. Knowing an intermediate hidden state, the two distant hidden states are d-separated. The information of the distant hidden state is irrelevant for inferring distributions since the intermediate hidden state already contains the information of the distant hidden state. This independance property allows for simple conditional distributions in hidden Markov models.

By definition:

$$p(t_k, w) = p(w_{k+1:n}|t_k, w_{1:k}) * p(t_k, w_{1:k}),\ \forall k \in \{1, .., n\}$$

. However with d-separation, $w_{k+1:n}$ is independant from $t_{1:k-1}$ and $w_{1:k}$ conditioned on t_k. Therefore,

$$p(t_k, w) = p(w_{k+1:n}|t_k) * p(t_k, w_{1:k}),\ \forall k \in \{1, .., n\}$$

. The forward path aims to calculate $p(t_k, w_{1:k}),\ \forall k \in \{1, .., n\}$ whereas the backward path aims to calculate $p(w_{k+1:n}|t_k),\ \forall k \in \{1, .., n\}$.

Forward path Given a sequence $w_{1:n}$, the goal of the forward path is to compute:

$$p(t_k, w_{1:k}) = p(t_k, w_1, w_2, ..., w_k),\ \forall k \in \{1, .., n\}$$

$$\begin{aligned}
\alpha_k(t_k) &= p(t_k, w_{1:k}) \\
&= \sum_{z_{k-1}=1}^{m} p(t_k, t_{k-1}, w_{1:k}) \\
&= \sum_{z_{k-1}=1}^{m} p(w_k|t_k, t_{k-1}, w_{1:k-1}) * p(t_k|t_{k-1}, w_{1:k-1}) * p(t_{k-1}, w_{1:k-1})
\end{aligned}$$

$w_{1:k-1}$ and t_{k-1} independant from w_k conditioned on t_k (d-separation):

$$\begin{aligned}\alpha_k(t_k) &= \sum_{z_{k-1}=1}^{m})p(w_k|t_k * p(t_k|t_{k-1}) * p(t_{k-1}, w_{1:k-1}) \\ &= \sum_{z_{k-1}=1}^{m} E_k(w_k) * T_{k,k-1} * \alpha_{k-1}(t_{k-1})\end{aligned}$$

Finally we showed (with $E_k(w)$ and $T_{k,k-1}$ the emission and transition matrices):

$$\alpha_k(t_k) = \sum_{z_{k-1}=1}^{m} E_k(w) * T_{k,k-1} * \alpha_{k-1}(t_{k-1})$$

Besides, we know that:

$$\alpha_1(t_1) = p(t_1, w_1) = p(t_1) * p(w_1|t_1) = \pi * E_1(w)$$

We can therefore apply the recursion to all the succession of events in the observed sequence in order to get $p(t_k, w_{1:k}),\ \forall k \in \{1, .., n\}$.

Backward path Given a sequence $w_{1:n}$, the goal of the backward path is to compute:

$$p(w_{k+1:n}|t_k),\ \forall k \in \{1, .., n-1\}$$

$$\begin{aligned}\beta_k(t_k) &= p(w_{k+1:n}|t_k) \\ &= \sum_{z_{k+1}=1}^{n} p(w_{k+2:n}|t_{k+1}, t_k, w_{k+1}) * p(w_{k+1}|t_{k+1}, t_k) * p(t_{k+1}|t_k)\end{aligned}$$

$w_{k+2:n}$ independant from t_k and w_{k+1} conditioned on t_{k+1} (d-separation):

$$\begin{aligned}\beta_k(t_k) &= \sum_{z_{k+1}=1}^{n} p(w_{k+2:n}|t_{k+1}) * p(w_{k+1}|t_{k+1}) * p(t_{k+1}|t_k) \\ &= \sum_{z_{k+1}=1}^{n} \beta_{k+1}(t_{k+1}) * E_{k+1}(w_{k+1}) * T_{k+1,k}\end{aligned}$$

Finally we showed (with $E_{k+1}(w_{k+1})$ and $T_{k+1,k}$ the emission and transition matrices):

$$\beta_k(t_k) = \sum_{z_{k+1}=1}^{n} \beta_{k+1}(t_{k+1}) * E_{k+1}(w_{k+1}) * T_{k+1,k}$$

Besides, we know that:

$$\beta_n(t_n) = 1$$

We can therefore apply the recursion to all the succession of events in the observed sequence in order to get $p(w_{k+1:n}|t_k),\ \forall k \in \{1, .., n-1\}$.

Viterbi algorithm for finding the most likely sequence of hidden states

The Viterbi algorithm aims to calculate the most likely sequence of hidden states for a sequence of observed variables $w_{1:n}$ given a HMM model ($\theta = (T, E, \pi)$). In a mathematical way, it aims to compute:

$$t* = \arg\max_{t_{1:n}} p(t_{1:n}|w_{1:n})$$

For simpler calculation, it computes $\arg\max_{t_{1:n}} p(t_{1:n}, w_{1:n})$ since

$$\arg\max_{t_{1:n}} p(t_{1:n}|w_{1:n}) = \arg\max_{t_{1:n}} p(t_{1:n}, w_{1:n})$$

. Indeed, $p(t_{1:n}|w_{1:n}) \propto p(t_{1:n}, w_{1:n})$ for a given sequence of observed variable $w_{1:n}$ since $p(t_{1:n}, w_{1:n}) = p(t_{1:n}|w_{1:n}) * p(w_{1:n})$.

As for the forward-backward algorithm, the goal is to establish a recursion between successive states.

$$\begin{aligned}
\mu_k(t_k) &= \max_{t_{1:k-1}} p(t_{1:k}, w_{1:k}) \\
&= \max_{t_{1:k-1}} p(w_k|t_k) * p(t_k|t_{k-1}) * p(t_{1:k-1}, w_{1:k-1}) \\
&= \max_{t_{1:k-1}} p(w_k|t_k) * p(t_k|t_{k-1}) * \max_{t_{1:k-2}} p(t_{1:k-1}, w_{1:k-1}) \\
&= \max_{t_{1:k-1}} E_k(w_k) * T_k * \mu_{k-1}(t_{k-1})
\end{aligned}$$

With $\mu_1(t_1) = p(t_1, w_1) = p(t_1) * p(w_1|t_1)$ (initial distribution * emission matrix).

In order to get the *argmax* out of the *max* calculation obtainable for each $k \in \{1, .., n\}$ with the recursion presented, the user must keep track of the sequence of t maximizing the joint probability $p(t_{1:k}, w_{1:k})$ at each step of the recursion.

5.2.3 Leverage the train set labels for automated feature creation

Amount and timing for modelling fraudulent and genuine behaviours

We stated earlier that experts could identify different types of fraudulent behaviours using clustering algorithm. These behaviours are based on the timing and the amount of transactions:

Cluster 1: Short sequences (2-4 transactions) of low amount transactions (less than 10 €).

Cluster 2: Burst (high number of transactions at the same merchant in a short amount of time) of medium amount transactions (10 - 100 €).

Cluster 3: Short sequences (3-5 transactions) of high amount transactions (more than 100 €).

Moreover, the timing of the transaction are used for transactions labelling when a fraud is detected. Previous transactions of the last 10 minutes are labelled by default as fraudulent, the transactions from the last day are situated in the grey zone and transactions occuring before the day of the fraudulent transactions detected are considered as genuine transactions.

Furthermore, the observation of the feature importance vectors of the random forests lead to the conclusion that the amount of the transaction is a predominant feature for credit card fraud detection.

Modelling the buying behaviours with respect to the amount and timing features could be interesting for fraud detection.

Model fraudulent and genuine behaviours

Credit card fraud detection is often considered as an anomaly detection task where one aims to spot abnormal behaviours which are supposed to be fraudulent behaviours. However, fraudsters aim to mimic genuine behaviours in order to remain undetected. Therefore the anomaly detection task is becoming harder. Moreover, we highlighted in section 5.2.3 that interviewed experts have been able to identify clusters of fraudster strategies.

The proposed feature engineering techniques aims to leverage the knowledge of the labels of the transactions of the training set in order to model genuine **and** fraudulent behaviour using hidden Markov models. Modelling the genuine behaviours is a classic anomaly detection scheme where the likelihood of a new

sequence of transactions is measured against historical honest sequences of transactions ([Chandola et al., 2012]).

The sequences used in order to model fraudulent behaviours are sequences of transactions of the training set with at least one fraudulent transaction. The rationale is that to have a risk of fraud, it is not enough for a new sequence to be far from usual transaction behavior but it is also expected for it to be relatively close to a risky behavior.

In the end, when assessing the fraudulence of a sequence of transaction, the classifier model would be able to know the distance between this sequence and the genuine behaviours and the distance between this sequence and the fraudulent behaviours. Double checking the distance between the observed sequence of transaction and the previously learned behaviours should decrease the number of false positive. As stated by [Pozzolo et al., 2017], this is a crucial issue since the investigators can only verify a limited number of alerts each day.

Multiple perspectives hidden Markov model based feature engineering

In addition to the descriptive aggregated features created by [Whitrow et al., 2008], we propose to create eight new HMM-based features. They quantify the similarity between the history of a transaction and eight distributions learned previously on set of sequences selected in a **supervised** way in order to model different perspectives.

In particular, we select three perspectives for modelling a sequence of transactions (see figure 5.4). A sequence (i) can be made only of genuine historical transactions or can include at least one fraudulent transaction in the history, (ii) can come from a fixed card-holder or from a fixed terminal, and (iii) can consist of amount values or of time-delta values (i.e. the difference in time between the current transaction and the previous one). We optimized the parameters of eight HMMs using all eight possible combinations (i-iii).

The three perspectives, namely genuine/fraudulent, card-holder/merchant and amount/time-delta, have been chosen based on the following assumptions:

- First, the features made with only genuine historical transactions should model the common behavior of card holders. The other four features are made with sequences of transactions with at least one fraudulent transaction and will model risky behaviours.
- The second perspective allows the model to take the point of view of the card-holder and the merchant which are the two actors involved in the credit

card transactions.

- The last perspective takes into account two important features for credit card fraud detection: the amount of a transaction and the elapsed time between two transactions. These features are strong indicators for fraud detection.

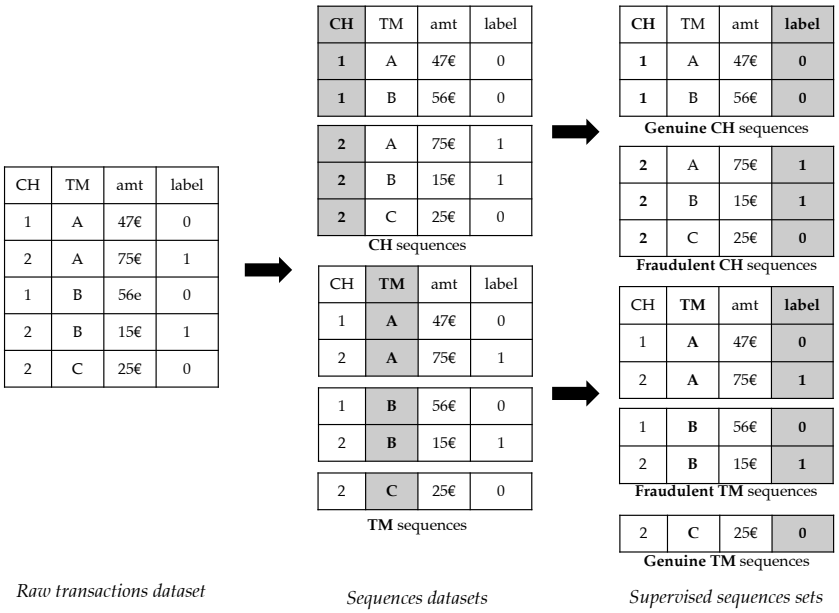


Figure 5.4: Enriching transaction data with HMM-based features calculated from multiple perspectives *(CH=Card-holder, TM=Terminal)*

In order to make the HMM model the genuineness and fraudulence of the card holders and the terminals, we create 4 training set (see figure 5.4) containing:

- sequences of transactions from **genuine credit cards** (without fraudulent transactions in their history).
- sequences of transactions from **compromised credit cards** (with at least one fraudulent transaction)
- sequences of transactions from **genuine terminals** (without fraudulent transactions in their history)
- sequences of transactions from **compromised terminals** (with at least one fraudulent transaction)

We then extract from these sequences of transactions the symbols that will be the observed variable for the HMMs. In our experiments, the observed variable can be either:

- the amount of a transaction.
- the amount of time elapsed between two consecutive transactions of a card-holder (time-delta).

At the end, we obtain 8 trained HMMs modeling 4 types of behaviour (genuine terminal behaviour, fraudulent terminal behaviour, genuine card-holder behaviour and fraudulent card-holder behaviour) for both observed variables (amount and time-delta).

Algorithm 1 Online: calculate likelihood of sequences of observed events

for tx_i in all transactions **do**
  for $perspective_j$ in perspectives combinations **do**
    $[tx_i, tx_{i-1}, tx_{i-2}] \leftarrow usersequence_{i,j}$
    $HMM_j \leftarrow HMM(usertype, signaltype, sequencetype)$
    $AnomalyScore \leftarrow log(p([tx_i, tx_{i-1}, tx_{i-2}]|HMM_j))$
  end for
end for

The HMM-based features described in this thesis (table 5.13) are the likelihoods that a sequence made of the current transaction and the two previous ones from this terminal/card holder is generated by each of these models.

As shown in algorithm 1, their calculation for each row (transaction) of the dataset involve fetching the HMM corresponding to each combination of perspective [3] in order to calculate the likelihoods that the sequence of three transactions have been generated by each HMM.

User	Feature	Genuine	Fraudulent
Card Holder	Amount	*HMM-cag*	*HMM-caf*
	Tdelta	*HMM-ctg*	*HMM-ctf*
Terminal	Amount	*HMM-tag*	*HMM-taf*
	Tdelta	*HMM-ttg*	*HMM-ttf*

Table 5.13: Set of 8 HMM-based features describing 8 combinations of perspectives

5.3 Summary

We propose an HMM-based feature engineering strategy that allows us to integrate sequential knowledge in the transactions in the form of HMM-based features. These HMM-based features enable a non sequential classifier (random

[3]The HMM properties in the algorithm 1 are: usertype: merchant/card-holder, signaltype: amount/timing, sequencetype: genuine/fraudulent

forest) to use sequential information for the classification.

The multiple perspective property of our HMM-based feature engineering strategy gives us the possibility to incorporate a broad spectrum of sequential information. In fact, we model the genuine and fraudulent behaviours of the merchants and the card-holders according to two features: the timing and the amount of the transactions. Moreover, the HMM-based features are created in a supervised way and therefore lower the need of expert knowledge for the creation of the fraud detection system.

Chapter 6

Experiments

In our work we proposed to generate history-based features using hidden Markov models. They quantify the similarity between an observed sequence and the sequences of past fraudulent or genuine transactions observed for the cardholders or the terminals.

In order to quantify how much the addition of multiple perspectives HMM-based features help the detection, we use the belgian credit card transactions dataset introduced in section 2.1.1.

The experimental protocol is described in section 6.1. In section 6.2.1 we present the increase in detection when adding HMM-based features using different classifiers (random forest classifier, Adaboost and logistic regression classifier).In section 6.2.2, we study hyperparameters influence: how much the window size and the number of hidden states affect the detection. Finally, in section 6.3 we compare different solutions to tackle the issue of structural missing values.

To ensure reproducibility, a source code for calculating and evaluating the proposed HMM-based features can be found at `https://gitlab.com/Yvan_Lucas/hmm-ccfd` .

6.1 Experimental Setup

6.1.1 Feature engineering and dataset partitioning

We use the Python library *hmmlearn*[1].

In order for the HMM-based features and the aggregated features to be comparable, we calculate terminal-centered aggregated features in addition to [Whitrow

[1] `https://github.com/hmmlearn/hmmlearn`

et al., 2008] card-holder centered aggregated features (table 6.1)

Feature	Signification
AGGCH1	# transactions issued by user in 24h.
AGGCH2	Amount spent by user in 24h.
AGGCH3	# transactions in the country in 24h.
AGGCH4	Amount spent in the country in 24h.
AGGTM1	# transactions in terminal in 24h.
AGGTM2	Amount spent in terminal in 24h.
AGGTM3	# transactions with this card type in 24h.
AGGTM4	Amount spent with this card type in 24h.

Table 6.1: Aggregated features centered on the card holders and the terminal

We split temporally the dataset in three different parts (see table 6.2: the training set, the validation set and the testing set. We chose to separate the time period corresponding to the training and validation set and the time period corresponding to the testing set with a gap of 7 days. The reason is that the in the real world fraud detection systems, human investigators have to verify the alerts generated by the classifiers. Since this process takes time, the ground truth is delayed by about one week. The transactions appearing in this gap of 7 days before the testing set are used in order to calculate the value of the aggregated and HMM-based features in the testing set but not for training the classifiers.

	Start Date	End Date
Training Set	01.03.2015	26.04.2015
Validation Set	27.04.2015	30.04.2015
Week gap	01.05.2015	07.05.2015
Testing Set	08.05.2015	31.05.2015

Table 6.2: Dataset splitting

6.2 Results

6.2.1 Improvement in fraud detection when using HMM-based features

We train classifiers using different feature sets in order to compare the efficiency of prediction when we add HMM-based features to the classification task for the face-to-face and e-commerce transactions. As stated in section 2.2.2, the precision-recall AUC metric is affected by class imbalance. Therefore, the difference in term of precision-recall AUC measured between the e-commerce and the

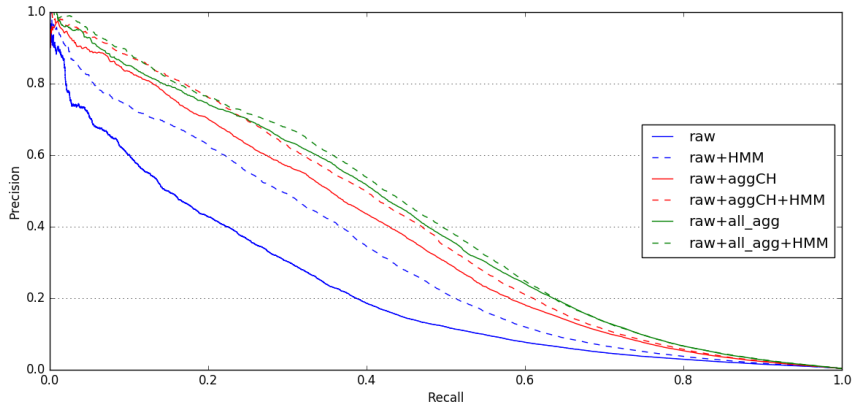
face-to-face transactions is due to the difference in the class imbalance between these datasets.

We tested the addition of our HMM-based features to several feature sets. We refer to the feature set **"raw+aggCH"** as the state of the art feature engineering strategy since it contains all the raw features with the addition of Whitrow's aggregated features ([Whitrow et al., 2008]). The feature groups we refer to are: the raw features (raw), the features based on the aggregations of card-holders transactions (aggCH), the features based on the aggregation of terminal transactions (aggTM), the proposed HMM-based features (HMM features).

In this section, the HMMs were created with 5 hidden states and the HMM-based features were calculated with a window-size of 3 (actual transaction + 2 past transactions of the card-holder and of the terminal). We showed in section 6.2.2 that the HMM hyperparameters (number of hiddens state and size of the window considered for the calculation of HMM-based features) did not change significantly the increase in Precision-Recall AUC observed.

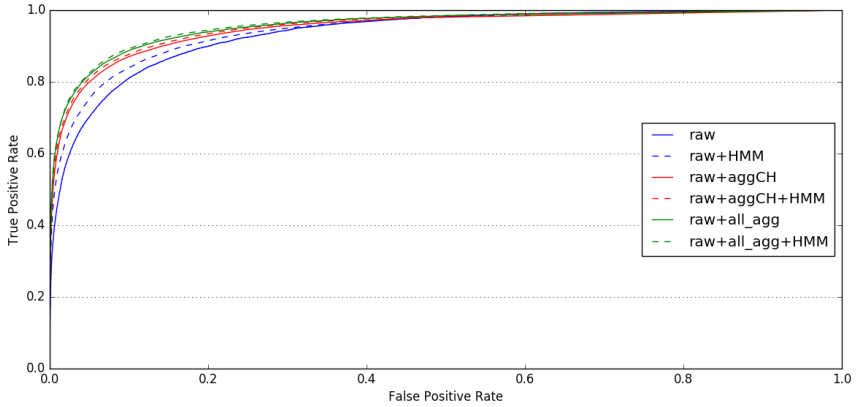
The figures show the precision recall curves and ROC curves with their AUC obtained by testing the efficiency of classifiers (random forest, logistic regression, Adaboost with shallow trees and adaboost with deep trees) trained with several feature sets on the transactions of the testing set. The AUC numbers corresponds to the average obtained on 3 different runs. The AUCs are very stable over the different runs and the standard deviation numbers are very low.

Addition to random forest classifiers



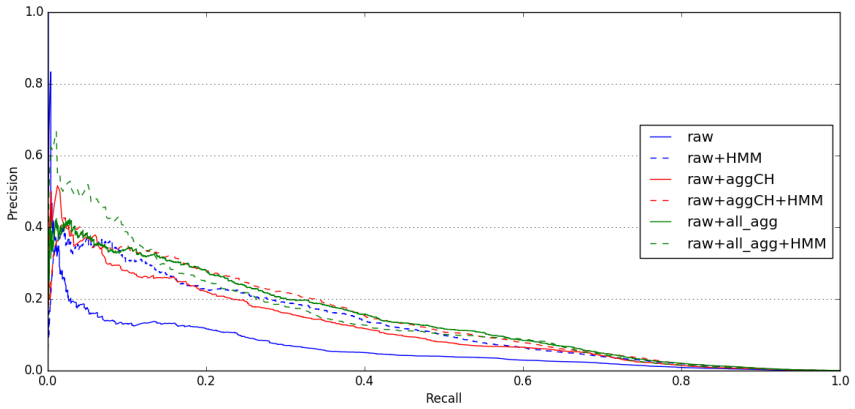
Feature set	no HMM-features	HMM-features	increase through HMMs
raw	0.212 ± 0.009	0.298 ± 0.0006	40.5%
raw+aggCH	0.343 ± 0.001	0.375 ± 0.0006	9.3%
raw+all agg	0.383 ± 0.002	0.397 ± 0.003	3.6%

Figure 6.1: Precision-recall curves for e-commerce transactions with random forest classifiers. *(Each color corresponds to a specific feature set, the line style corresponds to the presence or not of HMM-based features).*



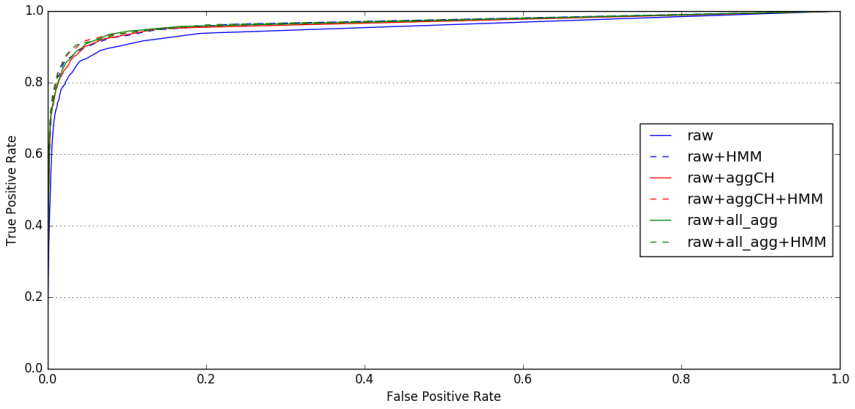
Feature set	no HMM-features	HMM-features	error decrease
raw	0.935 ± 0.0003	0.943 ± 0.0004	12.3%
raw+aggCH	0.950 ± 0.0001	0.955 ± 0.0006	10%
raw+all agg	0.957 ± 0.0002	0.960 ± 0.0005	6.9%

Figure 6.2: ROC curves for e-commerce transactions with random forest classifiers.



Feature set	no HMM features	HMM features	increase through HMMs
raw	0.082 ± 0.001	0.152 ± 0.001	85.4%
raw+aggCH	0.144 ± 0.001	0.170 ± 0.001	18.1%
raw+all agg	0.167 ± 0.001	0.177 ± 0.0006	6.0%

Figure 6.3: Precision-recall curves for face-to-face transactions with random forest classifiers. *(Each color corresponds to a specific feature set, the line style corresponds to the presence or not of HMM-based features).*



Feature set	no HMM-features	HMM-features	error decrease
raw	0.950 ± 0.0005	0.967 ± 0.0002	33%
raw+aggCH	0.964 ± 0.00005	0.967 ± 0.0008	9%
raw+all agg	0.967 ± 0.0004	0.969 ± 0.0003	6.1%

Figure 6.4: ROC curves for face-to-face transactions with random forest classifiers.

We tuned the random forest hyperparameters (table 6.3) through a grid search that optimized the Precision-Recall area under the curve on the validation set.

n-trees	n-features	min-samples-leaf	max depth
$\{300\}$	$\{1, 7, 13\}$	$\{1, 20, 40\}$	$\{4, None\}$

Table 6.3: Random forest grid search

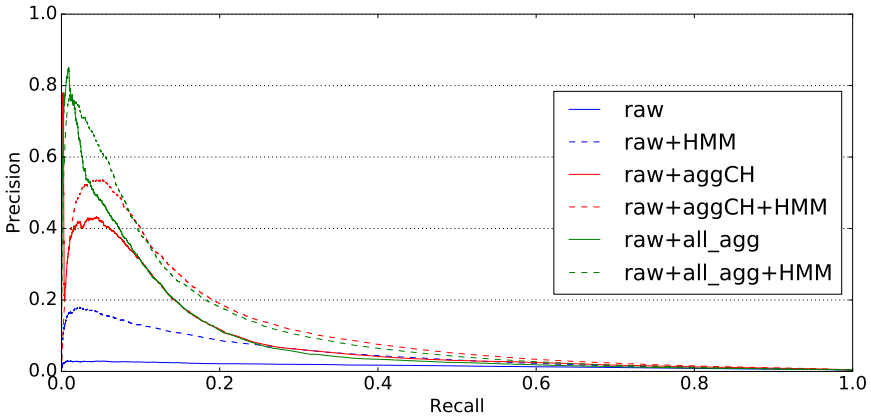
We can observe on figures 6.2 and 6.4 that the face-to-face and the e-commerce results both show a significant improvement in precision-recall AUC when adding the proposed HMM-based features to the raw feature set. For the e-commerce transactions, this improvement ranges from 3.6% for the best feature set without HMM-based features (raw+all-agg for the e-commerce transactions) to 40.5% for the worst feature set (raw for the e-commerce transactions). For the face-to-face transactions, this improvement ranges from 6.0% for the best feature set without HMM-based features (raw+all-agg for the face-to-face transactions) to 85.4% for the worst feature set (raw for the face-to-face transactions).

The relative increase in ROC AUC when adding HMM-based features is much lower due to the high value of these AUCs. However the decrease in misclassification done by the classifier is significant. It ranges from 6% to 33% for the face-to-face transactions and from 6.9% to 12.3% for the e-commerce transactions. The decrease in missclassification when adding HMM-based features to the state of the art feature engineering strategy ([Whitrow et al., 2008]) is of 10% for the e-commerce transactions and 9% for the face-to-face transactions.

By comparing the AUC of the curves raw+aggCH and raw+aggCH+HMM, we observe that adding HMM-based features to the state of the art feature engineering strategy introduced in the work of Whitrow & al. leads to an increase of 18.1% of the PR AUC for the face-to-face transactions and to an increase of 9.3% of the PR AUC for the e-commerce transactions.

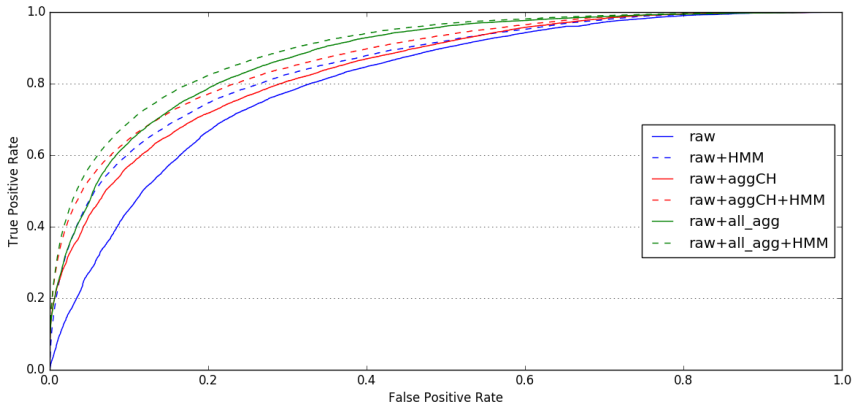
The relative increase in PR-AUC when adding terminal centered aggregated features to the feature set is of 16.0% for the face-to-face dataset. The relative increase in PR-AUC when adding terminal centered aggregated features to the feature set is of 11.7% for the e-commerce transactions. The terminal perspective is very valuable for the detection of fraudulent credit card transactions.

Addition to logistic regression classifiers



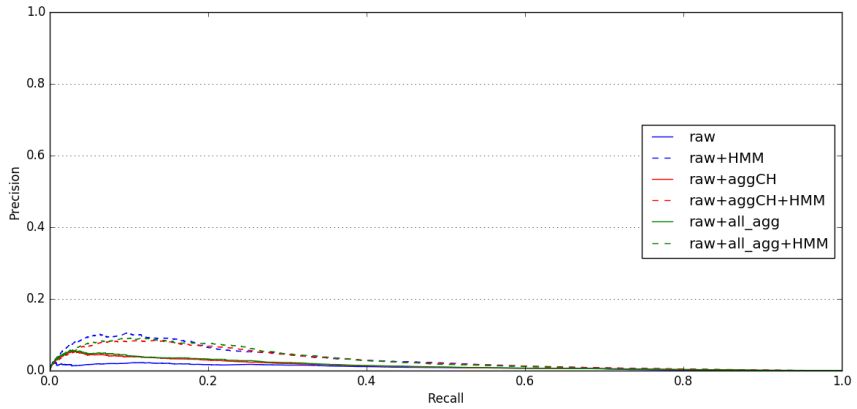
Feature set	no HMM-features	HMM-features	increase through HMMs
raw	0.015 ± 0.0001	0.049 ± 0.001	267%
raw+aggCH	0.092 ± 0.005	0.121 ± 0.003	31.5%
raw+all agg	0.096 ± 0.004	0.124 ± 0.005	29.2%

Figure 6.5: Precision-recall curves for e-commerce transactions with logistic regression. *(Each color corresponds to a specific feature set, the line style corresponds to the presence or not of HMM-based features.)*



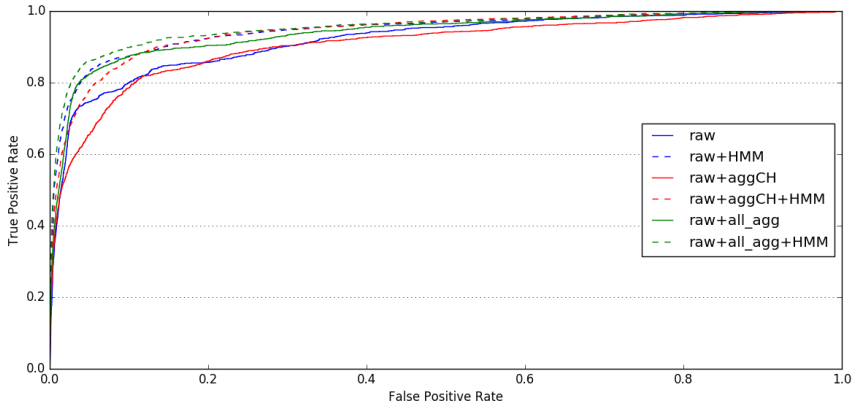
Feature set	no HMM-features	HMM-features	error decrease
raw	0.812 ± 0.002	0.856 ± 0.002	38.7%
raw+aggCH	0.856 ± 0.014	0.880 ± 0.009	55%
raw+all agg	0.867 ± 0.010	0.889 ± 0.008	66%

Figure 6.6: ROC curves for e-commerce transactions with logistic regression.



Feature set	no HMM features	HMM features	increase through HMMs
raw	0.0094 ± 0.0001	0.030 ± 0.002	219%
raw+aggCH	0.014 ± 0.001	0.0247 ± 0.02	76.4%
raw+all agg	0.0161 ± 0.0003	0.0315 ± 0.0007	95.6%

Figure 6.7: Precision-recall curves for face-to-face transactions with logistic regression. *(Each color corresponds to a specific feature set, the line style corresponds to the presence or not of HMM-based features).*



Feature set	no HMM-features	HMM-features	error decrease
raw	0.916 ± 0.002	0.943 ± 0.003	32.2%
raw+aggCH	0.923 ± 0.014	0.943 ± 0.002	26%
raw+all agg	0.933 ± 0.004	0.948 ± 0.005	32.5%

Figure 6.8: ROC curves for face-to-face transactions with logistic regression.

We tuned the logistic regression hyperparameters (table 6.4) through a grid search that optimized the Precision-Recall Area under the Curve on the validation set.

C parameter	penalty	tolerance for stopping
$\{1, 10, 100\}$	$\{l1, l2\}$	$\{10, 100\}$

Table 6.4: Logistic regression grid search

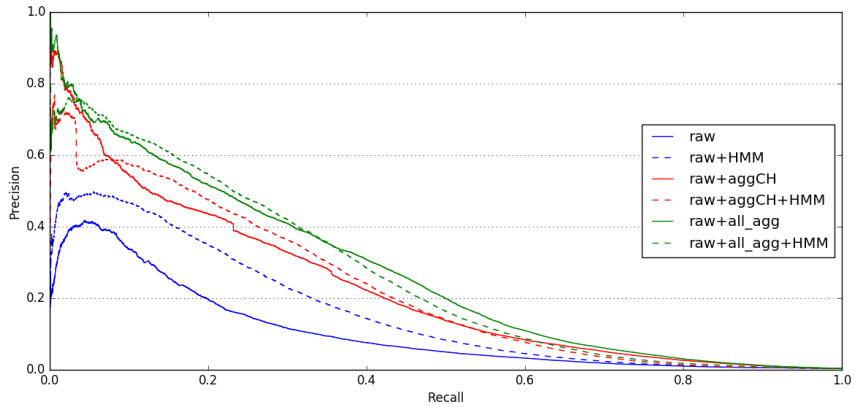
We can observe in figures 6.8 and 6.6 that the face-to-face and the e-commerce results both show a significant improvement in precision-recall AUC when adding the proposed HMM-based features to the raw feature set. For the e-commerce transactions, this improvement ranges from 29.2% for the best feature set without HMM-based features (raw+all-agg for the e-commerce transactions) to 267% for the worst feature set (raw for the e-commerce transactions). For the face-to-face transactions, this improvement ranges from 76.4% for the best feature set without HMM-based features (raw+aggCH for the face-to-face transactions) to 219% for the worst feature set (raw for the face-to-face transactions).

The relative increase in ROC AUC when adding HMM-based features is much lower due to the high value of these AUCs. However the decrease in misclassification done by the classifier is significant. It ranges from 26% to 32.5% for the face-to-face transactions and from 38.7% to 66% for the e-commerce transactions. The decrease in missclassification when adding HMM-based features to the state of the art feature engineering strategy ([Whitrow et al., 2008]) is of 26% for the face-to-face transactions and 55% for the e-commerce transactions.

By comparing the AUC of the curves raw+aggCH and raw+aggCH+HMM, we observe that adding HMM-based features to the state of the art feature engineering strategy introduced in the work of Whitrow & al. leads to an increase of 76.4% of the PR AUC for the face-to-face transactions and to an increase of 55% of the PR AUC for the e-commerce transactions.

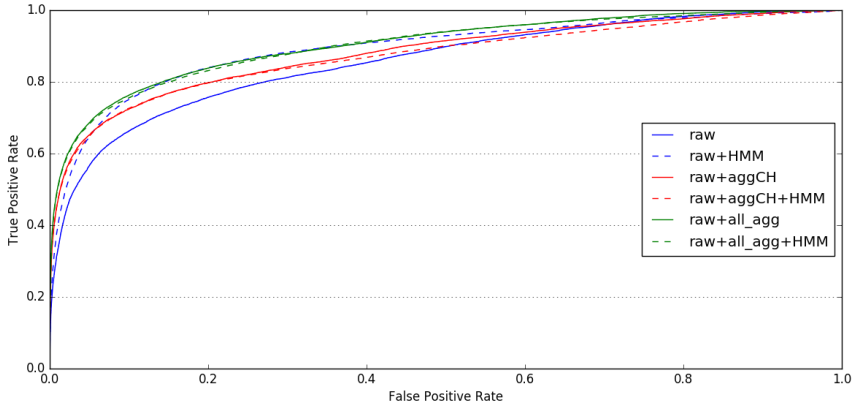
We observe a significant decrease in detection efficiency with logistic regression classifiers compared to random forest classifier. Logistic regression classifier is known to be weak to categorical features encoded with a label encoder (category transformed to numbers) since the label encoding implies an order between the categories that is irrelevant. The results obtained with logistic regression classifiers may be improved using one-hot encoding or frequency encoding like advised in some other credit card fraud detection project ([Pozzolo, 2015], [Fabry, 2019]), however the evolution observed when including the proposed feature engineering strategy to logistic regression classifier is consistent with what we observed using random forest classifier.

Addition to adaboost classifier



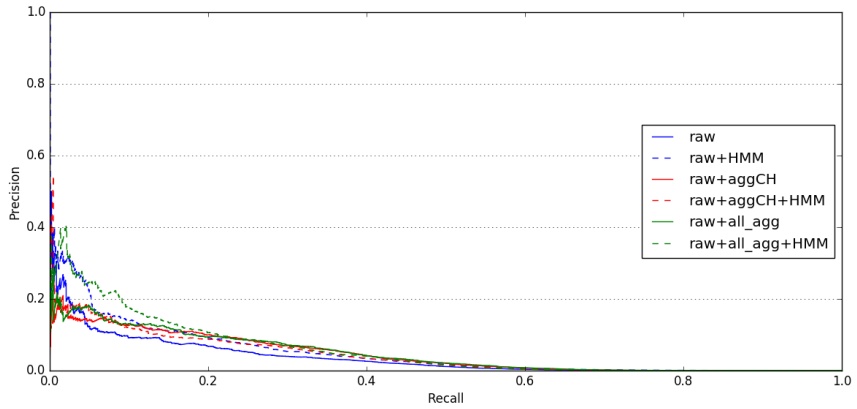
Feature set	no HMM-features	HMM-features	increase through HMMs
raw	0.103 ± 0.001	0.163 ± 0.007	58.3%
raw+aggCH	0.238 ± 0.009	0.247 ± 0.011	3.8%
raw+all agg	0.263 ± 0.004	0.264 ± 0.006	0.4%

Figure 6.9: Precision-recall curves for e-commerce transactions with Adaboost classifiers. *(Each color corresponds to a specific feature set, the line style corresponds to the presence or not of HMM-based features.)*



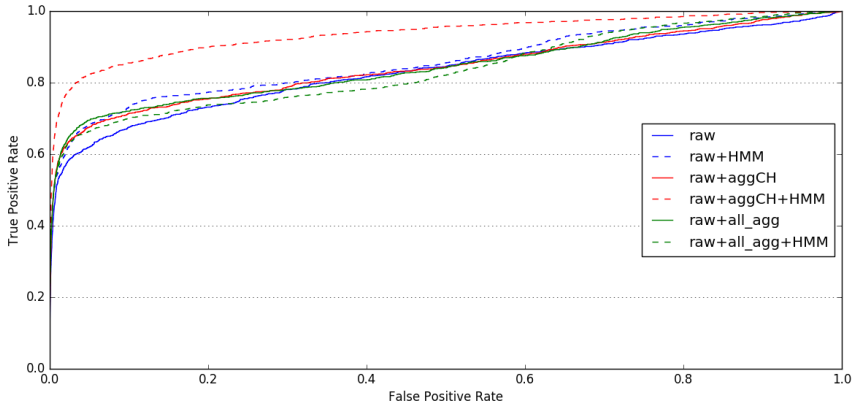
Feature set	no HMM-features	HMM-features	error decrease
raw	0.857 ± 0.003	0.874 ± 0.013	11.9%
raw+aggCH	0.873 ± 0.014	0.872 ± 0.004	-0.8%
raw+all agg	0.907 ± 0.006	0.905 ± 0.003	-2.1%

Figure 6.10: ROC curves for e-commerce transactions with Adaboost classifiers.



Feature set	no HMM features	HMM features	increase through HMMs
raw	0.034 ± 0.003	0.058 ± 0.009	70.6%
raw+aggCH	0.048 ± 0.002	0.079 ± 0.006	64.6%
raw+all agg	0.053 ± 0.003	0.061 ± 0.004	15.1%

Figure 6.11: Precision-recall curves for face-to-face transactions with Adaboost classifiers.*(Each color corresponds to a specific feature set, the line style corresponds to the presence or not of HMM-based features).*



Feature set	no HMM-features	HMM-features	error decrease
raw	0.825 ± 0.004	0.853 ± 0.018	16%
raw+aggCH	0.925 ± 0.024	0.937 ± 0.002	16%
raw+all agg	0.83 ± 0.001	0.83 ± 0.001	0%

Figure 6.12: ROC curves for face-to-face transactions with Adaboost classifiers.

We tuned the Adaboost hyperparameters (table 6.5) through a grid search that optimized the Precision-Recall Area under the Curve on the validation set.

tree numbers	learning rate	tolerance for stopping	max tree depth
$\{100, 400\}$	$\{0.1, 1, 100\}$	$\{10, 100\}$	$\{1, 4\}$

Table 6.5: Adaboost grid search

We can observe in figures 6.10 and 6.12 that the face-to-face and the e-commerce results both show a significant improvement in precision-recall AUC when adding the proposed HMM-based features to the raw feature set. For the e-commerce transactions, the improvement is significant for the less informative feature sets (58.3% improvement in PR AUC for the worst feature set). For the face-to-face transactions, this improvement ranges from 15.1% for the best feature set without HMM-based features (raw+all-agg for the face-to-face transactions) to 70.6% for the worst feature set (raw for the face-to-face transactions).

The relative increase in ROC AUC when adding HMM-based features is much lower due to the high value of these AUCs. However there is a small decrease in misclassification done by the classifier. It ranges from 0% to 16% for the face-to-face transactions and from -2.1% to 11.9% for the e-commerce transactions. The decrease in missclassification when adding HMM-based features to the state of the art feature engineering strategy ([Whitrow et al., 2008]) is of 0% for the e-commerce transactions and 16% for the face-to-face transactions.

By comparing the AUC of the curves raw+aggCH and raw+aggCH+HMM, we observe that adding HMM-based features to the state of the art feature engineering strategy introduced in the work of Whitrow & al. leads to an increase of 64.6% of the PR AUC for the face-to-face transactions and to an increase of 3.8% of the PR AUC for the e-commerce transactions.

We observed that shallow adaboost classifiers with a tree depth restricted to 1 as adviced in the literature ([Pedregosa et al., 2011]) lead to significantly worse detection of fraudulent transactions than deeper adaboost classifiers using trees of depth 4. The downside of deeper trees is the increased risk of overfitting of the adaboost classifiers since this ensemble method doesn't prevent overfitting like random forest do. Monitoring the number of iterations (tree added) from which the classifier starts to overfit could be done by tracking the PR-AUC on the validation set and on the training set. Due to adaboost design, the PR-AUC on the training set should increase for each tree added, however when the classifier overfits, the PR-AUC on the validation set would decrease when the classifier overfits: the classifier will start to learn particularities of the training set that are not relevant in other sets.

We can also note that the AUCs obtained with adaboost classifiers are larger

than the AUCs obtained with logistic regression classifier. This can be due to the fact that adaboost classifiers is composed by decision trees which are adapted to label encoded categorical features.

Conclusion

Overall, we can observe that, whatever the classifier used, the addition of features that describe the sequence of transactions, be it HMM-based features or Whitrow's aggregated features, increases a lot the Precision-Recall AUC on both e-commerce and face-to-face transactions. The addition of HMM-based features improves the prediction on both e-commerce and face-to-face transactions and allows the classifiers to reach the best levels of accuracy on both e-commerce and face-to-face transactions.

6.2.2 Robustness to hyperparameters changes

In order to understand if the feature engineering strategy is resilient to a change of the hyperparameters used for the construction of the HMM-based features, we constructed 9 sets of HMM-based features with different combinations of the HMM-based features hyperparameters. The hyperparameters considered are the number of hidden states in the HMMs and the size of the sequence of past transactions used for the calculation of the likelihood score.

We used random forest to measure the AUC obtained on the test set when adding different set of HMM-based features obtained with different combinations of hyperparameters to the raw feature set. We chose random forest classifiers for this evaluation since the precedent study showed that they were the best classifier for our credit card fraud detection task.

		Window size		
		3	5	7
Hidden states	3	0.280 ± 0.013	0.292 ± 0.015	0.295 ± 0.019
	5	**0.315 ± 0.008**	0.310 ± 0.006	0.297 ± 0.007
	7	0.307 ± 0.003	0.307 ± 0.004	0.296 ± 0.011

Table 6.6: PR-AUCs for E-commerce HMM hyperparameters ($raw = 0.203 \pm 0.005$)

We observe that, the combination of parameter {window size: 3, hidden states: 5} gives the best AUCs on average on 3 runs. However the standard deviation values (±) are too high to confidently say that a hyperparameters combination is better than all the others.

		Window size		
		3	5	7
Hidden	3	0.159 ± 0.020	0.135 ± 0.007	0.150 ± 0.013
states	5	**0.139 ± 0.006**	0.135 ± 0.012	0.123 ± 0.009
	7	0.129 ± 0.008	0.131 ± 0.009	0.124 ± 0.007

Table 6.7: PR-AUCs for Face-to-face HMM hyperparameters ($raw = 0.089 \pm 0.011$)

In a real world case, doing the hyperparameter search can be worth it but requires to calculate several set of HMM-based features. Overall we can conclude that the HMM-based feature engineering strategy is relatively stable over the hyperparameters.

We also observed that adding all 9 newly calculated HMM-based features sets provided a slight improvement on the face-to-face transactions (13.3%). We thought that adding HMM-based features with different hyperparameters could possibly bring different type of information. For example the window-size parameter could be seen as: description of short-term/mid-term/long-term history of the transaction. This small improvement wasn't observed on the e-commerce transactions (8.5% decrease).

6.3 Handling the missing value limitations

	e-commerce		face-to-face	
	# transactions	# frauds	# transactions	# frauds
All transactions	$5.4 * 10^6$	18472	$6.8 * 10^6$	2764
$History >= 3$	$4.5 * 10^6$	15650	$5.3 * 10^6$	1305
$History >= 7$	$3.4 * 10^6$	12493	$4.1 * 10^6$	859

Table 6.8: Number of transactions in the test set with different history constraints ($History >= 3$ means that all the transactions have at least 2 transactions in the past for the card-holder **and** for the terminal: we can build sequences of 3 transactions for both perspectives)

With the HMM feature engineering framework, we can calculate sets of HMM features for different window sizes. However, when the transaction history is not big enough we can't calculate the HMM-based features for it. Because of this limitation we could not consider around 20% of the transactions for the experiments of the results section (6.2.1) and around 40% of the transactions for the experiments of the hyperparameter section (6.2.2). There is a strong need to tackle the issue of structural missing values depending on the length of users sequences in order to integrate transactions with short history where part or all

of the HMM-based features couldn't have been calculated.

In this section we consider 9 sets of HMM based features obtained with a window size of 3, 5 and 7 for the card-holder or for the terminal. We therefore have 16 sets of transactions with different history constraints (terminal history: [0, 3, 5, 7] $*$ card-holder history: [0, 3, 5, 7]).

We try 3 missing values strategies in order to be able to combine predictions for different history constraints:

default0 a genuine default value solution where the HMM-based features that couldn't have been calculated for the current transactions are replaced by 0.

weighted PR a weighted sum of the predictions of the random forests specialized on the history constraints of the transaction, among 16 random forests each trained on one of the 16 possible history constraints. (For example, for a transaction with a terminal history of 6 and a card-holder history of 3 we will sum the predictions of the random forests [0,0], [0, 3], [3, 0], [3, 3], [5, 0], [5, 3][2]). Each random forest is weighted by it's efficiency on the validation set. We used PR AUCs values as weights for the random forests.

stacked RF a stacking approach where a random forest classifier is trained on the predictions of the 16 random forests specialized on the constraints (0 for missing prediction, when the transaction doesn't satisfy the constraints of the considered random forest). This approach has the second benefit that we might stack indivual classifiers thereby creating a more accurate new one.

Other approaches to handle missing values advise to generate them by modelling the distribution of the corresponding features. We considered that these solutions don't apply in our case since the missing values appear because there wasn't enough historical information to calculate the corresponding feature. We thought that replacing the value of a model based feature (HMM based features) using an other model (k-nearest neighbours or random forest) wasn't an appropriate solution.

We can observe in table 6.9 that the simplest solution (*default0*) allows for the best PR-AUCs with the best stability for both face-to-face and e-commerce transactions. This is also by far the fastest method since it needs to only train one random forest instead of 16 (resp. 17) for the *weighted PR* approach (resp.

[2]The first number describes the terminal history constraint, the second number describes the card-holder history constraint

	e-commerce PR-AUC	face-to-face PR-AUC
raw (no HMM-based features)	0.264 ± 0.002	0.058 ± 0.012
default0	0.320 ± 0.001	0.145 ± 0.004
weighted PR	0.275 ± 0	0.040 ± 0.002
stacked RF	0.317 ± 0.002	0.133 ± 0.021

Table 6.9: Fraud detection on the whole test set with different missing values strategies

stacked RF approach).

The *weighted PR* solution doesn't allow for satisfying results. However the *stacked RF* solution gives good PR-AUCs and presents interesting properties in order to combine different types of classifiers. It could be interesting to combine the predictions of an LSTM with the prediction of some HMM-based features enhanced random forest since these classifiers have been shown to not detect the same frauds in face-to-face transactions by [Jurgovsky et al., 2018].

Finding satisfying solutions to integrate transactions with structural missing value increases drastically the scope of the proposed framework (see table 6.8). The number of transactions that have satisfying characteristics to be classified using the HMM-based features framework is increased by 58% in the face-to-face context and by 65% in the e-commerce context with the addition of an appropriate missing value strategy. Moreover it adds an other perspective to the framework: HMM-based features calculated for a small (resp. big) window-size will characterize the short (resp. long) term history.

6.4 Summary

The terminal perspective is usually not used in credit card fraud detection and is shown in this paper to greatly help the detection for face-to-face and e-commerce transactions.

The results show an increase in the precision-recall AUC (18.1% for the face-to-face transactions and 9.3% for the e-commerce ones) due to the addition of our multi-perspective HMM-based features when compared to the state of the art feature engineering strategies. We also showed that this increase is robust to the hyperparameters of the HMMs.

Moreover, we briefly compare the most common solutions in order to handle the structural missing value problem that happens when the history of the card-holder (resp. terminal) is too short.

Chapter 7

Perspectives and conclusions

7.1 Perspectives

7.1.1 Frequential decomposition for transactions clustering

Further work could be done in order to characterize the buying behaviour (either restricted to the sequences of the card-holders/merchants or for the whole dataset) from a frequential point of view. Indeed, Fourier transforms allow users to decompose any signal (for example the sequence of amount spent, or the sequence of timings) into sub-signals for a deeper understanding of the underlying behaviour.

For example, a research direction to follow would be to find a way to separate different buying habits with the help of Fourier decomposition or Wavelet decomposition in order to associate and compare incoming transactions with transactions of the same type. This could allow to divide a set of examples into several specialized sets containing only transactions of the same type (everyday transactions, holidays transactions, christmas transactions, Back to School transactions, etc). These sets could afterwards be leveraged for training more specialized classifiers, or optimize the parameters of hidden Markov models in order to model particular types of buying behaviours in the context of the multiple perspective HMM-based feature engineering described in this thesis.

In the end, knowledge coming from the frequency domain could bring to light helpful information in order to differenciate buying behaviours and maybe increase the detection of fraudulent credit card transactions.

7.1.2 Interpretation of classifier decision

As highlighted in the research questions, interpretability is a crucial challenge of machine learning. Indeed, some of the most popular models are black boxes because of their number of parameters: it is not humanly possible to understand the decision process of neural networks or random forests by printing the millions of weights linking neurons or by looking at each decision tree one by one.

Besides, machine learning is taking more and more space in everyday life and therefore more and more crucial decisions are taken with the help of machine learning algorithms in domains such as medical diagnosis, fraud detection, autonomous driving, etc. Being able to explain models decisions is very important in case of model failure: an insurance would want to know why an autonomous car has crashed, a relative would want to know why the data driven medical diagnosis model failed to discover the disease it was supposed to discover.

There already exists approaches in order to approach model decisions:

- First, some models contain a small enough number of parameters to be humanly understandable. One of the asset of the multiple perspective HMM-based feature engineering highlighted in this thesis is that since HMM contain few parameters (63 for a gaussian HMM with 7 hidden states). This small number of parameter allows to output the trained model and analyze its decision: for example we could observe that genuine sequences of amount modelled by the HMM are steadier than fraudulent sequences of amount. Moreover, such small models are not able to strongly fit to the data due to their reduced number of parameter. This is an asset and a drawback: the model has to compress the observed behaviours and retain only a strong abstraction of it, therefore it is more prone to underfitting (see section 2.1.2.

- Studying the features importances of tree based classifiers also enables to identify which attributes are more relevant for the classifier decision.Authors such as [Karax et al., 2019] used feature importance vectors for characterizing the evolution of classifiers decision. However, we observed that the feature importance varies for different types of classifiers for the same learning task: boosting trees feature importance vectors and random forest feature importance vectors were very different for the credit card fraud detection task for example. Moreover, feature importance has biases towards categorical features with a big number of categories according to [Altmann et al., 2010].

- Recently some model agnostic methods have been proposed in order to

reach model decision. [Pastor and Baralis, 2019] proposed to approach the classifiers decisions with rules that are produced in the neighbourhood of the model decision function. The set of rules produced can afterwards be used in order to have a coarse-grained view of the model decision. [Altmann et al., 2010] proposed to use permutation importance instead, which measure the importance of a feature for model decision by the decrease in model prediction when shuffling the elements of this feature. If the decrease is important when shuffling a considered feature, that means that this feature was important for the classifier decision and that having it non informative (because shuffled) cripple the classifier. Others authors such as [Zintgraf et al., 2017] aimed to understand neural networks decisions by using prediction difference analyis[1] in order to highlight which portions of an image are relevant in an image classification task (ImageNet).

However, one question that is still to be adressed for model interpretability is: "Towards which class prediction a given attribute move the decision function?" Indeed, the aforementioned methods quantify the importance of each attribute for the decision but aren't able to explain in which direction each attribute moves the decision: some questions such as: "Are transactions with a big amount more likely to be predicted as fraudulent transactions?" can not be answered by these methods.

Interpretability of machine learning model is a trendy topic and a motivating research direction. We could measure the importance of this question during the meetings with our industrial partner Worldline: it is harder to trust a model decision if its decision is obfuscated.

7.2 Conclusions

At the beginning of this work we introduced the stakes of credit card fraud detection and how it is managed in practice. More precisely, we introduced the concepts of real time and near real time fraud detection in one hand, and the double authentication routines performed for e-commerce and for face-to-face transactions.

However, an axis of progression that was still subject to explorations is: how to describe and afterwards integrate the contextual information relative to each credit card transaction? Indeed, fraudulent credit card transactions are very similar to genuine credit card transactions and the knowledge of the buying habits of

[1] Similarly to permutation importance, the prediction difference analysis estimates the relevance of a feature by measuring how the prediction changes when the feature is unknown

the concerned card-holder may help distinguish the genuine transactions from the fraudulent transactions within his sequence. Moreover, the buying behaviours are subject to change over seasons and affect the ability of a fraud detection system to efficiently detect fraudulent transactions.

The main contributions of this work lead to the construction of a framework in order to model in an effective way sequential behaviours from multiple perspectives (see chapter 5). Typical approaches for considering sequences in a credit card fraud detection context aim to extract descriptive statistics from genuine card-holder sequences of transactions.

- The first contribution of this work is to consider terminal perspective in addition to card-holder perspective. It has been shown in section 5.1.2 that genuinely considering the terminal perspective leads to an average of 10% increase in precision-recall AUC.

- Second, we proposed to model not only genuine behaviours but also fraudulent behaviours (for both card-holder and terminal perspectives). The rationale is that to have a risk of fraud, it is not enough for a new sequence to be far from a genuine buying behaviour but it is also expected for it to be relatively close to a risky behaviour. Assessing the distance from a fraudulent behaviour would allow the classifier to reduce the number of false alerts where the buying behaviour diverge from an usual behaviour without being similar to a risky behaviour.

- Finally, we observed that the state of the art fraud detection feature engineering approaches aimed to extract descriptive statistics from sequences in order to enrich transaction information. We highlighted in section 5.1.3 that these descriptive statistic would fail to reveal differencies between two very different types of card-holders sequences: steady transactions amounts vs oscillating transactions amounts. We chose to use a generative model (hidden Markov model) in order to describe the temporal dependencies between the transactions of a sequence. The choice of a generative model was motivated because we wanted to really represent the sequential behaviour whereas a discriminant model would represent the sequential relationship between a target variable y and a set of attributes x.

Using this multiple perspective HMM-based feature engineering strategy, we could increase the precision-recall AUC of 18.1% for face-to-face credit card fraud detection and 9.3% for the e-commerce credit card fraud detection. Moreover, we could show that this feature engineering strategy is relevant for various types of classifiers (random forest, logistic regression and Adaboost) and robust to hyperparameters choices made for constructing the features. The main drawback

of this approach is the fact that there may be structural missing values when the history is too short to calculate the value of such HMM-based features. Several solutions were benchmarked in order to provide a feature engineering that can be applied for all the transactions, including the ones with short history. We can imagine building similar HMM-based features in any supervised task that involves a sequential dataset.

The other contribution of this work was to propose a method for qualifying the seasonal covariate shift in any temporal dataset (see chapter 4). We built a distance matrix between the days of the credit card transactions dataset by evaluating the ability of a classifier to predict the day of each transaction. We could show that, within the belgian credit card transactions dataset, the covariate shift followed a calendar-based pattern. When integrating this covariate shift knowledge in the credit card fraud detection process, we observed a slight increase in precision-recall AUC.

The multiple perspectives HMM-based feature engineering strategy was proven to be an asset in order to increase classifiers performance for the credit card fraud detection task. It is even more interesting since approaches to characterize sequences of multidimensional categorical attributes are less prevalent than methods to characterize sequences of continuous signals in the litteratiure. Moreover, this method is not specific to credit card fraud detection and could be exported to other application domains such as medical diagnosis for example. We can imagine that modelling past evolution of the symptoms of a patient in order to compute a risk score of his situation could be relevant for decision takers, be it machine learning models or doctors.

Bibliography

[Abdallah et al., 2016] Abdallah, A., Maarof, M. A., and Zainal, A. (2016). Fraud detection system: A survey. *Journal of Network and Computer Applications.*

[Acuna and Rodriguez, 2004] Acuna, E. and Rodriguez, C. (2004). The treatment of missing values and its effect on classifier accuracy. *Classification, clustering and data mining applications.*

[Alaiz-Rodriguez and Japkowicz, 2008] Alaiz-Rodriguez, R. and Japkowicz, N. (2008). Assessing the impact of changing environments on classifier performance. *Conference of the Canadian Society for Computational Studies of Intelligence, Springer.*

[Ali et al., 2019] Ali, M. A., Azad, M. A., Centeno, M. P., Hao, F., and van Moorsel, A. (2019). Consumer-facing technology fraud: Economics, attack methods and potential solutions. *Future Generation Computer Systems (volume 100).*

[Altmann et al., 2010] Altmann, A., Tolosi, L., Sander, O., and Lengauer, T. (2010). Permutation importance: a corrected feature importance measure. *Bioinformatics 26(10).*

[Baesens et al., 2015] Baesens, B., Vlasselaer, V. V., and Verbeke, W. (2015). Fraud analytics using descriptive, predictive, and social network techniques: A guide to data science for fraud detection. *Book published by Wiley.*

[Bahnsen et al., 2015] Bahnsen, A. C., Aouada, D., Stojanovic, A., and Ottersten, B. (2015). Detecting credit card fraud using periodic features. *IEEE 14th International Conference on Machine Learning and Applications.*

[Bahnsen et al., 2016] Bahnsen, A. C., Ouada, D., Stojanovic, A., and Ottersten, B. (2016). Feature engineering strategies for credit card fraud detection. *Expert Systems with Applications.*

[Barddal and Enembreck, 2019] Barddal, J. P. and Enembreck, F. (2019). Learning regularized hoeffding trees from data streams. *34th ACM/SIGAPP Symposium on Applied Computing (SAC2019)*.

[Batista et al., 2000] Batista, G., Carvalho, A. C., and Monard, M. C. (2000). Applying one-sided selection to unbalanced datasets. *MICAI 2000: Advances in Artificial Intelligence*.

[Batista et al., 2004] Batista, G., Prati, R., and Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*.

[Bauer and Kohavi, 1999] Bauer, E. and Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning 36(1-2)*.

[Baum, 1972] Baum, L. E. (1972). An inequality and associated maximization technique. *Statistical estimation for probabilistic functions of Markov processes*.

[Bayer, 2015] Bayer, J. S. (2015). Learning sequence representations. *PhD thesis, Technische Universitat Munchen*.

[Bengio and Frasconi, 1995] Bengio, Y. and Frasconi, P. (1995). An input output hmm architecture. *Advances in neural information processing systems*.

[Bengio et al., 1994] Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*.

[Best and Fisher, 1979] Best, D. J. and Fisher, N. I. (1979). Efficient simulation of the von mises distribution. *Journal of the Royal Statistical Society*.

[Bhattacharyya et al., 2011] Bhattacharyya, S., Jha, S., Tharakunnel, K., and Westland, J. C. (2011). Data mining for credit card fraud: A comparative study. *Decision support systems*.

[Bickel et al., 2007] Bickel, S., Brückner, M., and Scheffer, T. (2007). Discriminative learning for differing training and test distributions. *Proceedings of the 24th International conference on Machine learning*.

[Bifet and Gavalda, 2009] Bifet, A. and Gavalda, R. (2009). Adaptive learning from evolving data streams. *International Symposium on Intelligent Data Analysis*.

[Blattberg et al., 2008] Blattberg, R. C., Kim, B.-D., and Neslin, S. A. (2008). Database marketing: analyzing and managing customers. *Book published by Springer*.

[Bolton and Hand, 2001] Bolton, R. J. and Hand, D. J. (2001). Unsupervised profiling methods for fraud detection. *Credit Scoring and Credit Control*.

[Bolton and Hand, 2002] Bolton, R. J. and Hand, D. J. (2002). Statistical fraud detection: a review. *Statistical science*.

[Bontemps et al., 2016] Bontemps, L., Cao, V. L., McDermott, J., and Le-Khac, N.-A. (2016). Collective anomaly detection based on long short-term memory recurrent neural networks. *International Conference on Future Data and Security Engineering*.

[Breiman, 1996] Breiman, L. (1996). Bagging predictors. *Machine learning 24(2)*.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine learning 45(1)*.

[Carcillo et al., 2018] Carcillo, F., Pozzolo, A. D., Borgne, Y.-A. L., Caelen, O., Mazzer, Y., and Bontempi, G. (2018). Scarff; a scalable framework for streaming credit card fraud detection with spark. *Information Fusion*.

[Chandola et al., 2012] Chandola, V., Banerjee, A., and Kumar, V. (2012). Anomaly detection for discrete sequences: A survey. *IEEE Transactions on Knowledge and Data Engineering*.

[Chawla, 2005] Chawla, N. V. (2005). Data mining for imbalance datasets: an overview. *Data mining and knowledge discovery handbook*.

[Chawla et al., 2002] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research 16*.

[Chawla et al., 2004] Chawla, N. V., Japkowicz, N., and Kotcz, A. (2004). Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*.

[Chawla et al., 2003] Chawla, N. V., Lazarevic, A., Bowyer, K. W., and Hall, L. O. (2003). Smoteboost: Improving prediction of the minority class in boosting. *Knowledge Discovery in Databases: PKDD*.

[de Fortuny et al., 2014] de Fortuny, E. J., Stankova, M., Moeyersoms, J., Minnaert, B., Provost, F., and Martens, D. (2014). Corporate residence fraud detection. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*.

[Delamaire et al., 2009] Delamaire, L., Abdou, H., and Pointon, J. (2009). Credit card fraud and detection techniques: a review. *Banks and Bank systems*.

[Dhok, 2012] Dhok, S. S. (2012). Credit card fraud detection using hidden markov model. *International Journal of Soft Computing and Engineering.*

[Dietterich, 2002] Dietterich, T. G. (2002). Machine learning for sequential data: A review. *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR).*

[Dorj et al., 2013] Dorj, E., Chen, C., and Pecht, M. (2013). A bayesian hidden markov model-based approach for anomaly detection in electronic systems. *IEEE Aerospace conference.*

[Drummond and Holte, 2003] Drummond, C. and Holte, R. (2003). C4.5, class imbalance, and cost sensitivity: why undersampling beats over-sampling. *Workshop on Learning from Imbalanced Datasets II.*

[Elkan, 2001] Elkan, C. (2001). The foundations of cost-sensitive learning. *International Joint Conference on Artificial Intelligence.*

[Ergen et al., 2017] Ergen, T., Mirza, A. H., and Kozat, S. S. (2017). Unsupervised and semi-supervised anomaly detection with lstm neural networks. *arXiv preprint.*

[Estabrooks et al., 2004] Estabrooks, A., Jo, T., and Japkowicz, N. (2004). A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence.*

[Europol, 2012] Europol (2012). Situation report: payment card fraud in the european union. perspective of law enforcement agencies. *https://www.europol.europa.eu/publications-documents/situation-report-payment-card-fraud-in-european-union.*

[Fabry, 2019] Fabry, R. (2019). equensworldline fraud risk management. *Credit Card Fraud Detection Workshop.*

[Fan et al., 1999] Fan, W., Stolfo, S. J., Zhang, J., and Chan, P. K. (1999). Adacost: misclassification cost-sensitive boosting. *ICML.*

[Fawcett and Provost, 1997] Fawcett, T. and Provost, F. (1997). Adaptive fraud detection. *Data mining and knowledge discovery.*

[Forrest et al., 1999] Forrest, S., Warrender, C., and Pearlmutter, B. (1999). Detecting intrusions using system calls: Alternate data models. *Proceedings of the 1999 IEEE ISRSP.*

[Fossi and Gianini, 2019] Fossi, L. and Gianini, G. (2019). Managing a pool of rules for credit card fraud detection by a game theory based approach. *Future Generations Computer Systems.*

[Fu et al., 2016] Fu, K., Cheng, D., Tu, Y., and Zhang, L. (2016). Credit card fraud detection using convolutional neural networks. *International Conference on Neural Information Processing.*

[Gao et al., 2007] Gao, J., Fan, W., Han, J., and Yu, P. (2007). A general framework for mining concept-drifting data streams with skewed distributions. *Proceedings of the 2007 SIAM International Conference.*

[Ghosh and Reilly, 1994] Ghosh, S. and Reilly, D. L. (1994). Credit card fraud detection with a neural-network. *IEEE Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences.*

[Gomes et al., 2017] Gomes, H. M., Bifet, A., Read, J., Barddal, J. P., Enembreck, F., Pfharinger, B., Holmes, G., and Abdessalem, T. (2017). Adaptive random forest for evolving data stream classification. *Machine Learning 106(9).*

[Gornitz et al., 2015] Gornitz, N., Braun, M., and Kloft, M. (2015). Hidden markov anomaly detection. *32nd International Conference on Machine Learning.*

[Graves, 2012] Graves, A. (2012). Supervised sequence labelling with recurrent neural networks. *Springer vol. 385.*

[Graves and Jaitly, 2014] Graves, A. and Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks. *Proceedings of the 31st International Conference on Machine Learning.*

[Gretton et al., 2009] Gretton, A., Smola, A., Huang, J., Schmittful, M., and Borgwardt, K. (2009). Covariate shift by kernel mean matching. *Dataset shift in machine learning 3(4).*

[Guo and Berkhahn, 2016] Guo, C. and Berkhahn, F. (2016). Entity embeddings of categorical variables. *CoRR.*

[Han et al., 2005] Han, H., Wang, W.-Y., and Mao, B.-H. (2005). Borderline-smote: a new oversampling method in imbalanced data sets learning. *Advances in intelligent computing, Springer.*

[He et al., 2008] He, H., Bai, Y., Garcia, E., and Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. *IEEE International Joint Conference on Neural Networks.*

[Hoare, 2019] Hoare, J. (2019). What is a decision tree? *www.displayr.com/what-is-a-decision-tree/ (last visited: August 2019).*

[Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation 9.*

[Hoens et al., 2011] Hoens, R. T., Chawla, N. V., and Polikar, R. (2011). Heuristic updatable weighted random subspaces for non-stationary environments. *IEEE 11th International Conference on Data Mining.*

[Hofmeyr et al., 1998] Hofmeyr, S. A., Forrest, S., and Somayaji, A. (1998). Intrusion detection using sequences of system calls. *Journal of Computer Security.*

[Hyunsoo et al., 2005] Hyunsoo, K., Golub, G. H., and Haesun, P. (2005). Missing value estimation for dna microarray gene expression data: local least squares imputation. *Bioinfomatics 21(2).*

[Japkowicz and Stephen, 2002] Japkowicz, N. and Stephen, S. (2002). The class imbalance problem: a systematic study. *Intelligent data analysis.*

[Jha et al., 2012] Jha, S., Guillen, M., and Westland, J. (2012). Employing transaction aggregatin strategy to detect credit card fraud. *Expert Systems with Applications.*

[Jurgovsky et al., 2018] Jurgovsky, J., Granitzer, M., Ziegler, K., Calabretto, S., Portier, P.-E., He-Guelton, L., and Caelen, O. (2018). Sequence classification for credit-card fraud detection. *Expert systems with applications.*

[Karax et al., 2019] Karax, J. A. P., Malucelli, A., and Barddal, J. P. (2019). Decision tree-based feature ranking in concept drifting data streams. *34th ACM/SIGAPP Symposium on Applied Computing (SAC2019).*

[Kelly et al., 1999] Kelly, M. G., Hand, D. J., and Adams, N. M. (1999). The impact of changing populations on classifier performance. *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining ACM.*

[Kingsford and Salzberg, 2008] Kingsford, C. and Salzberg, S. L. (2008). What are decision trees? *Nat Biotechnol 26(9).*

[Klinkenberg, 2003] Klinkenberg, R. (2003). Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis vol 8.*

[Kolter and Maloof, 2003] Kolter, J. Z. and Maloof, M. A. (2003). Dynamic weighted majority: A new ensemble method for tracking concept drift. *Third IEEE International conference on data mining.*

[Kotsiantis, 2007] Kotsiantis, S. (2007). Supervised machine learning: A review of classification techniques. *Informatica*.

[Kubat and Matwin, 1997] Kubat, M. and Matwin, S. (1997). Addressing the curse of imbalanced training sets: one-sided selection. *ICML*.

[Kukar and Kononenko, 1998] Kukar, M. and Kononenko, I. (1998). Cost-sensitive learning with neural networks. *ECAI*.

[Lafferty et al., 2001] Lafferty, J., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*.

[Laleh and Azgomi, 2009] Laleh, N. and Azgomi, M. A. (2009). A taxonomy of frauds and fraud detection techniques. *International Conference on Information Systems, Technology and Management*.

[Lane and Brodley, 1998] Lane, T. and Brodley, C. E. (1998). Approaches to online learing and concept drift for user identification in computer security. *KDD*.

[Lichtenwalter and Chawla, 2009] Lichtenwalter, R. N. and Chawla, N. V. (2009). Adaptive methods for classification in arbitrarily imbalanced and drifting data streams. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*.

[LifeCLEF, 2018] LifeCLEF (2018). Lifeclef 2018 bird challenge. *https://www.imageclef.org/node/230 (visited during 08/2019)*.

[Liu et al., 2009] Liu, X.-Y., Wu, J., and Zhou, Z.-H. (2009). Exploratory undersampling for class-imbalance learning. *Systems, Man, and Cybernetics, part B: Cybernetics*.

[Lucas et al., 2019a] Lucas, Y., Portier, P.-E., Laporte, L., Calabretto, S., Caelen, O., He-Guelton, L., and Granitzer, M. (2019a). Multiple perspectives hmm-based feature engineering for credit card fraud detection. *34th ACM/SIGAPP Symposium on Applied Computing (SAC2019)*.

[Lucas et al., 2019b] Lucas, Y., Portier, P.-E., Laporte, L., Calabretto, S., Caelen, O., He-Guelton, L., and Granitzer, M. (2019b). Towards automated feature engineering for credit card fraud detection using multi-perspective hmms. *Future Generations Computer Systems Special Issue on: Data Exploration in the Web 3.0 Age*.

[Lucas et al., 2019c] Lucas, Y., Portier, P.-E., Laporte, L., Calabretto, S., He-Guelton, L., Oblé, F., and Granitzer, M. (2019c). Dataset shift quantification for credit card fraud detection. *Artificial intelligence and knowledge engineering (AIKE2019).*

[Malhotra et al., 2015] Malhotra, P., Vig, L., Shroff, G., and Agarwal, P. (2015). Long short term memory networks for anomaly detection in time series. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning.*

[Mani and Zhang, 2003] Mani, I. and Zhang, I. (2003). knn approach to unbalanced data distributions: a case study involving information extraction. *Proceedings of Workshop on Learning from Imbalanced Datasets.*

[McCallum et al., 2000] McCallum, A., Freitag, D., and Pereira, F. (2000). Maximum entropy markov models for information extraction and segmentation. *17th International Conference on Machine Learning (ICML 2000).*

[Mikolov et al., 2011] Mikolov, T., Deoras, A., Kombrink, S., Burget, L., and Cernocky, J. (2011). Empirical evaluation and combination of advanced language modeling techniques. *12th Annual Conference of the International Speech Communication Association.*

[Moreno-Torres et al., 2012] Moreno-Torres, J., Raeder, T., Alaiz-Rodriguez, R., Chawla, N. V., and Herrera, F. (2012). A unifying view on dataset shift in classification. *Pattern recognition.*

[Moreno-Torres et al., 2013] Moreno-Torres, J. G., Llora, X., Goldberg, D. E., and Bhargava, R. (2013). Repairing fractures between data using genetic programming-based feature extraction: A case study in cancer diagnosis. *Information Sciences 222.*

[Noghani and Moattar, 2015] Noghani, F. F. and Moattar, M. H. (2015). Ensemble classification and extended feature selection for credit card fraud detection. *Journal of AI and Data Mining.*

[Oba et al., 2003] Oba, S., Sato, M. A., Takemasa, I., Monden, M., Matsubara, K. I., and Ishii, S. (2003). A bayesian missing value estimation method for gene expression profile data. *Bioinfomatics 19(16).*

[Pastor and Baralis, 2019] Pastor, E. and Baralis, E. (2019). Explaining black box models by means of local rules. *34th ACM/SIGAPP Symposium on Applied Computing (SAC2019).*

[Patidar et al., 2011] Patidar, R., Sharma, L., et al. (2011). Credit card fraud detection using neural network. *International Journal of Soft Computing and Engineering (IJSCE)*.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*.

[Power, 2011] Power, D. (2011). Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *Journal of machine learning technology (1)*.

[Pozzolo, 2015] Pozzolo, A. D. (2015). Adaptive machine learning for credit card fraud detection. *PhD Thesis*.

[Pozzolo et al., 2017] Pozzolo, A. D., Boracchi, G., Caelen, O., Alippi, C., and Bontempi, G. (2017). Credit card fraud detection: a realistic modeling and a novel learning strategy. *IEEE transactions on neural networks and learning systems*.

[Pozzolo et al., 2014] Pozzolo, A. D., Caelen, O., and Borgne, Y. L. (2014). Learned lessons in credit card fraud detection from a practitioner perspective. *Expert systems with applications*.

[Quinlan, 1996] Quinlan, R. J. (1996). Bagging, boosting and c4.5. *AAAI/IAAI Vol. 1*.

[Rabiner and Juang, 1991] Rabiner, L. R. and Juang, B. H. (1991). Hidden markov models for speech recognition. *Technometrics*.

[Rashid et al., 2016] Rashid, T., Agrafiotis, I., and Nurse, J. R. C. (2016). A new take on detecting insider threats: exploring the use of hidden markov models. *ACM CCS International Workshop on Managing Insider Security Threats (MIST)*.

[Robinson and Aria, 2018] Robinson, W. N. and Aria, A. (2018). Sequential fraud detection for prepaid cards using hidden markov model divergence. *Expert Systems with Applications*.

[Rumelhart et al., 1986] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Cognitive modeling*.

[Russac et al., 2018] Russac, Y., Caelen, O., and He-Guelton, L. (2018). Embeddings of categorical variables for sequential data in fraud context. *International Conference on Advanced Machine Learning Technologies and Applications.*

[Shapley, 1953] Shapley, L. S. (1953). A value for n-person games. *Annals of Mathematics Study, Princeton University Press.*

[Shimodaira, 2000] Shimodaira, H. (2000). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference 90(2).*

[Shirazi and Vasconcelos, 2010] Shirazi, H. M. and Vasconcelos, N. (2010). Risk minimization, probability elicitation and cost-sensitive svms. *ICML.*

[Srivastava et al., 2008] Srivastava, A., Kundu, A., Sural, S., and Majumdar, A. K. (2008). Credit card fraud detection using hidden markov model. *IEEE Transactions on dependable and secure computing.*

[Stekhoven and Buhlmann, 2011] Stekhoven, D. J. and Buhlmann, P. (2011). Missforest-non-parametric missing value imputation for mixed-type data. *Bioinfomatics 28(1).*

[Storkey, 2009] Storkey, A. (2009). When training and test sets are different: characterizing learning transfer. *Dataset shift in machine learning.*

[Sugiyama et al., 2007] Sugiyama, M., Krauledat, M., and Mazller, K.-R. (2007). Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research 8.*

[Sun et al., 2007] Sun, Y., Kamel, M. S., Wong, A. K., and Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition.*

[Troyanskaya et al., 2001] Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. (2001). Missing value estimation methods for dna microarrays. *Bioinfomatics 17(6).*

[Visa and Ralescu, 2005] Visa, S. and Ralescu, A. (2005). Issues in mining imbalanced data sets: a review paper. *Proceedings of the sixteen midwest artificial intelligence and cognitive science conference.*

[Viterbi, 1967] Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions Information Theory, vol. IT-13.*

[Vlasselaer et al., 2015] Vlasselaer, V. V., Bravo, C., Caelen, O., Eliassirad, T., Akoglu, L., Snoeck, M., and Baesens, B. (2015). Apate: A novel approach for automated credit card transactions fraud detection using network-based extensions. *Decision support systems.*

[Wang et al., 2003] Wang, K., Zhou, S., Fu, C. A., and Yu, J. (2003). Mining changes of classification by correspondance tracing. *Proceedings of the 2003 SIAM International COnference on Data Mining. Society for Industrial and Applied Mathematics.*

[Wang et al., 2009] Wang, S., Tang, K., and Yao, X. (2009). Diversity exploration and negative correlation learning on imbalanced data sets. *International Joint Conference on Neural Networks.*

[Webb and Ting, 2005] Webb, G. I. and Ting, K. M. (2005). On the application of roc analysis to predict classification performance under varying class distributions. *Machine Learning 58.*

[Weiss and Provost, 2001] Weiss, G. M. and Provost, F. (2001). The effect of class distribution on classifier learning: an empirical study. *Rutgers Univ.*

[Whitrow et al., 2008] Whitrow, C., Hand, D. J., Juszczak, P., Weston, D. J., and Adams, N. M. (2008). Transaction aggregatin as a strategy for credit card fraud detection. *Data Mining and Knowledge Discovery 18(1).*

[Widmer and Kubat, 1996] Widmer, G. and Kubat, M. (1996). Learning in the presence of context drift and hidden contexts. *Machine Learning 23.*

[Wu and Chang, 2003] Wu, G. and Chang, E. Y. (2003). Class-boundary alignment for imbalanced dataset learning. *ICML workshop on learning from imbalanced data sets II.*

[Yan et al., 2003] Yan, R., Liu, Y., Jin, R., and Hauptmann, A. (2003). On predicting rare classes with svm ensembles in scene classification. *IEEE International Conference on Acoustics, Speech, and Signal Processing.*

[Zadrozny et al., 2003] Zadrozny, B., Langford, J., and Abe, N. (2003). Cost-sensitive learning by cost-proportionate example weighting. *Data Mining ICDM, IEEE.*

[Zintgraf et al., 2017] Zintgraf, L., Cohen, T., Adel, T., and Welling, M. (2017). Visualising deep neural network decisions: prediction difference analysis. *arXiv preprint.*

UNIVERSITÉ DE LYON

INSA

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : LUCAS DATE de SOUTENANCE : 12/2019

Prénoms : Yvan, François, Marcel

TITRE :

Credit Card Fraud Detection using Machine Learning with Integration of Contextual Knowledge

NATURE : Doctorat Numéro d'ordre : 2019LYSEIXXXX

Ecole doctorale : INFOMATHS

Spécialité : Informatique

RESUME :

La détection de fraude par carte de crédit présente plusieurs caractéristiques qui en font une tâche difficile. Tout d'abord, les attributs décrivant une transaction ignorent les informations séquentielles qui se sont avérées très pertinentes pour la détection des fraudes à la carte de crédit.
Des solutions pour intégrer des informations séquentielles au sein des attributs transactionnels existent dans la littérature. La stratégie principale consiste à créer un ensemble d'attributs qui sont des statistiques descriptives obtenues en agrégeant les séquences de transactions des titulaires de carte (somme du montant, nombre de transactions, etc.). Nous avons utilisé cette méthode comme méthode de référence pour la détection des fraudes à la carte de crédit. Cependant, cette stratégie de prétraitement des données a soulevé plusieurs questions de recherche. Tout d'abord, nous avons supposé que ces statistiques descriptives ne pouvaient pas décrire complètement les propriétés séquentielles des motifs temporels frauduleux et non frauduleux et que la modélisation des séquences de transactions pouvait être bénéfique pour la détection de la fraude. De plus, la création de ces fonctionnalités agrégées est guidée par des connaissances expertes, tandis que la modélisation de séquences pourrait être automatisée grâce aux labels de classe disponibles pour les transactions passées. Enfin, ces attributs agrégées sont des estimations ponctuelles pouvant être complétées par une description multi-perspective du contexte de la transaction (en particulier du point de vue du vendeur).
Nous avons proposé une stratégie pour la création d'attributs basés sur des modèles de Markov cachés (HMM) caractérisant la transaction par différents points de vue. Cette stratégie permet d'intégrer un large spectre d'informations séquentielles dans les attributs des transactions. En fait, nous modélisons les comportements authentiques et frauduleux des commerçants et des détenteurs de cartes selon deux caractéristiques univariées: la date et le montant des transactions. De plus, les attributs basées sur les HMM sont créées de manière supervisée, réduisant ainsi le besoin de connaissances expertes pour la création du système de détection de fraude. En fin de compte, notre approche à perspectives multiples basée sur des HMM permet un prétraitement automatisé des données pour modéliser les corrélations temporelles afin de compléter et éventuellement remplacer les stratégies d'agrégation de transactions afin d'améliorer l'efficacité de la détection.
Des expériences menées sur un vaste ensemble de données de transactions de cartes de crédit issu du monde réel (46 millions de transactions effectuées par des porteurs de carte belges entre mars et mai 2015) ont montré que la stratégie proposée pour le prétraitement des données basé sur les HMM permet de détecter davantage de transactions frauduleuses quand elle est combinée à la stratégie de prétraitement des données de référence basées sur des connaissance expertes pour la détection de fraude à la carte de crédit. La stratégie poposée de construction d'attributs à l'aide des HMM ne devrait pas remplacer la stratégie de réference pour le prétraitement des données car la meilleure détection est obtenue grâce de la combinaison de ces stratégies de prétraitement des données.

MOTS-CLÉS : machine learning, fraud detection, hidden Markov model, sequence modelling, feature engineering, dataset shift

Laboratoire (s) de recherche : LIRIS DRIM

Directeur de thèse: Prof. Dr. Sylvie Calabretto & Prof. Dr. Michael Granitzer

Président de jury : Prof. Dr. Chantal Soulé-Dupuy

Composition du jury : Prof. Dr. Chantal Soulé Dupuy, Prof. Dr. Eric Gaussier, Prof. Dr. Mathias Lux, Dr. Gabriele Gianini, Prof. Dr. Sylvie Calabretto, Prof. Dr. Michael Granitzer, Dr. Pierre-Edouard Portier, Dr. Léa Laporte