N°5 | STEPWISE

IA EN PRODUCTION

CYCLE DE VIE ET DÉRIVE DES MODÈLES

Quantmetry

RÉDACTEURS

Nicolas Bousquet, Florian Canderlé, Antoine Charlet, Olivier Denti, Benjamin Habert, Issam Ibnouhsein, Charlotte LeDoux, Gaultier Le Meur, Gill Morisse, Nicolas Peltre, Mathieu Ringoot

SOMMAIRE

PRÉFACE

L'Intelligence Artificielle (IA) est généralement définie comme un ensemble de concepts et de technologies mises en oeuvre en vue de réaliser des machines capables de reproduire le comportement humain. Ce livre blanc s'intéresse plus particulièrement à l'IA non symbolique, en l'occurrence **le Machine Learning et son sous-ensemble le Deep Learning**, que l'on peut définir comme l'ensemble des d'algorithmes capables d'apprendre à résoudre un problème, depuis les données, sans être explicitement programmés.
Par essence, ces modèles vivent dans un environnement en constante évolution car les données utilisées, les concepts appris et les environnements accueillant l'algorithme vont évoluer dans le temps. Concrètement, cela signifie que la mise à disposition du modèle aux utilisateurs finaux n'est pas la dernière étape, et qu'il faut mettre en place des procédures de maintien en conditions opérationnelles. C'est ce qu'on appelle **le cycle de vie des modèles.**

Dans la continuité de notre livre blanc de 2018 sur l'interprétabilité des modèles "IA explique toi", cette problématique nous confronte à nouveau au sujet de la confiance que l'on peut accorder à un modèle d'IA. Cette fois-ci, nous l'aborderons sous l'angle de la fiabilité et de la durabilité des algorithmes implémentés. Cette problématique est plus que jamais d'actualité puisque de nombreuses entreprises exploitant des modèles en production se retrouvent face à des implémentations qui échouent encore à susciter l'adhésion de leurs utilisateurs. Pire, certains modèles peuvent devenir inutilisables peu après après leur mise en production...

On peut noter trois types d'enjeux autour du cycle de vie :

› **Data science** : Comment construire le modèle le plus robuste possible et s'assurer qu'il reste robuste à de nouvelles situations? Quelle doit être la politique de réentrainement et d'analyse des performances dans le temps de façon non biaisée sachant que mon modèle impacte ma cible ? Quels garde-fous mettre en oeuvre dans un contexte de prise de décision automatisée ?

› **Data engineering** : Comment s'assurer du déploiement en continu de mon modèle dans des situations de maintenance ou d'évolutions programmées ? Quel versionnage et monitoring des modèles mettre en place? Comment sécuriser au mieux les flux de données ?

› **Organisationnel** : Comment piloter la gestion des modèles dans l'entreprise ? Comment s'assurer de leur bonne utilisation, que les risques ont été correctement définis ? Que les indicateurs suivis sont adéquats ?

Ainsi, le cycle de vie des modèles définit un ensemble de processus permettant d'organiser la vie du modèle dans l'entreprise et de répondre à des problématiques terrain souvent observées parmi nos clients :

› Comment convaincre les différents interlocuteurs (métier, direction, DSI) de la valeur ajoutée et de la faisabilité du modèle d'IA, sa robustesse étant un pré-requis ?

› Comment réagir en cas de problème avec un modèle ? Notamment comment faire en sorte que des équipes très hétérogènes (data scientists/ engineer, DSI, métiers) puissent travailler ensemble pour maintenir le service avec suffisamment de réactivité ?

› Comment assurer le ROI d'un algorithme reposant sur des données qui sont susceptibles d'évoluer au cours du temps pour des raisons variées ?

› Comment gérer la multiplication des modèles d'IA dans l'entreprise et quels sont les risques associés à cette transformation ?

La gestion du cycle de vie des modèles et de leur gouvernance est par conséquent un enjeu stratégique pour construire des modèles utilisables après leur industrialisation, et dont la réponse nécessite une réelle transformation des processus de l'entreprise. L'objectif de ce livre blanc est d'apporter des éléments de réponse concrets sur ce sujet.

Bonne lecture !

Florian Gardin
Senior Data Scientist

Aurélia Nègre
Senior Data Scientist

REMERCIEMENTS

La rédaction de ce livre blanc a été menée par un ensemble de contributeurs et relecteurs que nous remercions chaleureusement, en s'appuyant sur les retours d'expérience des entreprises en pointe sur les sujets Big Data.

Nous remercions donc ces entreprises et particulièrement nos interlocuteurs pour les interviews, qui ont permis d'enrichir le contenu des différents chapitres par leur retour d'expérience :

Amazon Web Services . OLIVIER **CRUCHANT,** *Machine Learning Architect*

Databricks . ARDUINO **CASCELLA,** *Solutions Architect*

Telecom ParisTech . JACOB **MONTIEL,** *Chercheur et contributeur principal de scikit-multiflow*

BPCE . EMMANUEL **SOUQUE,** *Responsable pilotage projets à la Direction des Risques*

Veepee . YOUSSEF **BENCHEKROUN**, *Manager et Lead Data Scientist*

Bleckwen . NINA **BERTRAND,** *Senior Data Scientist*

MAIF . CHRISTELLE **LUGUET,** *Chargée d'action développement des réseaux*

FRÉDÉRIC **DE JAVEL,** *Chef de projet data science*

Alkemics . PIERRE **ARBELET,** *Lead Data Scientist*

1. désigne le résultat d'une activité humaine sous la forme d'un bien ou d'un service.
2. qui possède toutes les qualités techniques et toutes les caractéristiques de fonctionnement d'un nouveau produit.
3. une personne qui exerce la royauté
4. courir
5. système permettant de ralentir

1. INDUSTRIALISATION DES MODÈLES : OÙ EN SOMMES-NOUS EN 2019 ?

***Contributeurs** : Olivier Denti, Benjamin Habert, Guillaume Hochard, Issam Ibnouhsein, Charlotte Ledoux, Gill Morisse, Mathieu Ringoot,*

2 ans après notre livre blanc, *"Y'a-t-il une vie après les POCs ? Réussir l'industrialisation du Big Data"*, nous avons pu constater, lors des missions réalisées chez nos clients et ce, dans l'ensemble des secteurs d'activité, une évolution du marché et des acteurs, caractérisée par la montée en compétences des équipes en interne et la maturité des entreprises en matière de stratégie, d'organisation et de gouvernance autour de leurs initiatives data.

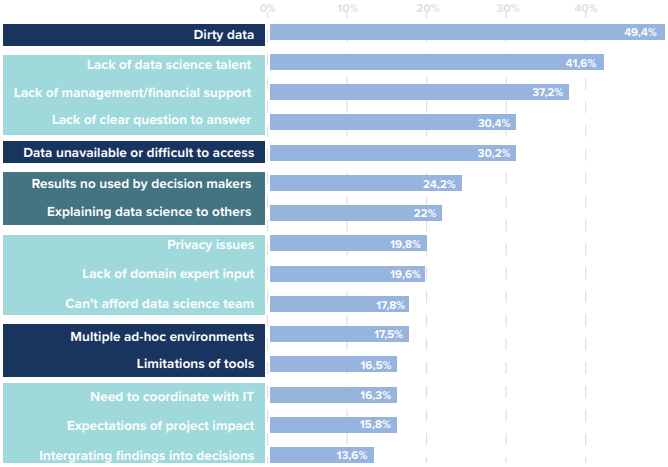
La plupart des projets d'IA commencent traditionnellement par une phase de PoC (*Proof of Concept*), cherchant à démontrer sur un périmètre réduit de données qu'un algorithme pourra être entraîné à résoudre une tâche particulière. Aujourd'hui, nous faisons le constat que moins de 20 % des projets d'IA passent de la phase de PoC à la phase de mise en production.

Se déroulant sur une période de temps relativement réduite (typiquement 1 à 2 mois), un écueil souvent observé est de penser que la mise en production du PoC sera aisée et rapide. En effet, cette étape peut être vue comme n'étant qu'une simple mise à disposition du code sur la plateforme de production où le modèle développé aura accès aux données et à l'infrastructure cible sur laquelle s'exécuter. De fait, la phase d'industrialisation est souvent sous-estimée dans sa durée et sa complexité.

A. ETAT DES LIEUX : POURQUOI DE NOMBREUX MODÈLES NE PARTENT TOUJOURS PAS EN PRODUCTION?

À la fin de l'année 2017, la plateforme communautaire Kaggle réalisait une photographie des difficultés rencontrées lors des projets data, mettant en lumière des freins aux initiatives data en matière de qualité de données ou de ressources humaines, en particulier la difficulté de constituer des équipes de data pluridisciplinaires pouvant embrasser l'ensemble des compétences requises dans un projet de data science. Aujourd'hui, même si le marché de l'emploi sur le secteur de l'intelligence artificielle reste très compétitif, les équipes data se sont étoffées et sont montées en compétences grâce à l'expérience acquise en cours de PoC et projets pilotes, déplaçant de fait le centre de gravité de ces difficultés vers la mise en production de ces initiatives.

QUELS FREINS AUX PROJETS DATA ? (16000 RÉPONDANTS)



http://blog.kaggle.com/2017/10/30/introducing-kaggles-state-of-data-science-machine-learning-report-2017

De notre retour d'expérience, les principaux freins observés aujourd'hui portent sur les périmètres de la gouvernance de la donnée, la gouvernance projet et l'humain (sponsors et soutien, l'adhésion des équipes métier, la complexité de mise en oeuvre par rapport à la maturité des acteurs), les aspects techniques (modularité et scalabilité du code et coûts de run trop élevés au regard de la plateforme disponible et de la modélisation choisie), les aspects réglementaires (privacy) ou bien encore un ROI non démontré ou trop faible.

B. PRINCIPAUX FREINS À L'INDUSTRIALISATION

LA GOUVERNANCE DE LA DONNÉE

Souvent mal comprise, considérée comme un facteur de coût au ROI approximatif, la gouvernance de la donnée représente pourtant le principal écueil des organisations traditionnelles sur la route de l'industrialisation. Rappelons que la gouvernance des données a pour vocation principale de permettre à l'ensemble des métiers, data scientists compris, d'accéder à une donnée maîtrisée, sécurisée et surtout exploitable.

Sur chacun de ces trois aspects, les freins qui persistent ne sont pas dûs à l'absence d'outils techniques mais plutôt à une incompréhension fondamentale sur les problématiques culturelles et humaines que soulève l'émergence d'une société où la donnée joue un rôle de plus en plus stratégique.

Car contrairement aux croyances qui confèrent à l'intelligence artificielle le pouvoir de tirer de la valeur de données brutes, massives et déstructurées, la data science requiert une donnée de qualité, qualifiée et décrite par les experts métiers. Afin de bien comprendre les difficultés qui se présentent à nos ingénieurs, voici quelques illustrations :

Manque de connaissance sur la qualité réelle de la donnée : en phase de PoC, les data scientists travaillent souvent sur des échantillons très qualitatifs et traités à la main. Alors que tous les feux semblent au vert, ils échouent en phase pilote car le périmètre d'étude est en fait parfaitement incomplet. Sur un projet consistant à détecter l'attrition client dans le retail, on se rend compte tardivement que seule la donnée d'une partie des magasins est accessible.

Manque d'une interprétation homogène des données et de coordination des projets : pour deux projets consommant la même donnée, les ingénieurs développent des pipelines de récupération de la donnée qui donnent des résultats différents. Chacun a utilisé des définitions métiers et des règles de cleansing différentes.

Manque de maîtrise de la qualité des données : nos ingénieurs, en cours de projet peuvent découvrir que la manière d'enrichir la donnée à la source a

évolué depuis les premières phases exploratoires. Par exemple, une direction commerciale décide de ne plus enrichir une donnée clé pour un modèle ou une refonte de l'ERP intervient en pleine phase d'industrialisation d'un modèle. Dans le même registre, une fois mis en production, un modèle peut connaître une dégradation que les ingénieurs ont du mal à expliquer faute de relais éclairés au sein du métier.

Manque de gouvernance des modèles de données : c'est un écueil souvent rencontré dans l'industrie où les schémas de données peuvent varier selon les usines, les machines, les produits au fil de leur renouvellement. Un projet pour lequel on avait calculé un ROI global peut vite se retrouver isolé sans possibilité de passer à une échelle globale.

Manque d'outils de partage de la connaissance : c'est un écueil majeur dans la capacité de l'organisation à créer de la valeur de manière durable. La multiplication de PoC ou d'expérimentations, menés par de multiples équipes internes ou externes, engendre un besoin aigu de capitalisation. La mise en oeuvre d'outils de type data dictionary permet alors d'assurer la transmission de la connaissance métier acquise sur la donnée entre équipes sur plusieurs projets consommant la même donnée ou sur un même projet qu'on souhaite déployer en production. Ces outils permettront également de déployer de nouvelles organisations permettant de monitorer la qualité de la donnée au long cours.

Manque de certitudes sur la conformité réglementaire. Il n'est pas rare qu'au moment de passer à l'industrialisation, on se rende compte que des données personnelles ne sont, en fait, pas utilisables en l'état et nécessitent la récupération du consentement. Au delà des délais supplémentaires considérables, le résultat final est de modifier fondamentalement la composition des échantillons de données utilisées pour entraîner les modèles.

Toutes ces illustrations permettent de comprendre qu'au delà des problématiques techniques, il est nécessaire de mettre en place une organisation pour capitaliser sur la connaissance de la donnée, pour contrôler et garantir une exploitabilité constante de la donnée. Et c'est d'autant plus important, qu'au delà de l'intelligence artificielle, les gains que procurent la gouvernance de la donnée sont essentiels pour les organisations au sens large :

- harmoniser les processus métiers

- faciliter le partage d'une information fiable entre les collaborateurs
- maîtriser la sécurité des données et leur conformité réglementaire
- valoriser la donnée comme un asset fondamental de l'entreprise

LA GOUVERNANCE PROJET

Compétence et autonomie des équipes, phasage/orchestration des tâches.
Ceci inclut la bonne identification des parties prenantes au projet afin de couvrir l'ensemble des phases et domaine compétence requis. Un des freins que nous rencontrons souvent est une réticence des acteurs IT à déployer sur leur infrastructure des codes qu'ils ne maîtrisent pas. Un moyen d'éviter cet écueil est d'intégrer l'IT très tôt dans le projet, idéalement dans le cadrage afin qu'ils aient connaissance du sujet et puisse anticiper une industrialisation. Cet exemple vaut pour tout acteur et souligne l'importance d'une communication large d'un sujet dès la phase de cadrage.

Une des conditions clés de l'industrialisation est de s'assurer que les interactions nécessaires entre les différentes partie prenante du projet sont planifiées et que les ressources temps sont bien allouées, de la phase de collecte/sélection de données jusqu'à la phase de *test & learn* et d'intégration dans le SI, en s'assurant qu'il n'y a pas de collision ou conflits d'agenda pour délivrer les projets propres à chaque entité.

Les différentes tâches et livrables sont clairement définis et chaque parties prenantes connaît sa contribution et responsabilité tout au long du projet. La mise en place d'un RACI (matrice de rôles et responsabilités, voir chapitre 3) est une bonne pratique à généraliser afin d'offrir la visibilité nécessaire sur les périmètres de chacun.

LES FACTEURS HUMAINS

L'industrialisation n'est pas qu'une question de processus et développement technique, elle implique également des personnes ce qui ajoute un facteur humain.
Les notions d'acculturation et de conduite du changement sont essentielles à l'industrialisation. Les utilisateurs finaux doivent être intégrés dans le processus

d'industrialisation que cela soit au niveau des recettes ou des phases de *test & learn*. Un suivi fin de l'utilisation lors cette dernière permet de mesurer l'appropriation métier de la solution et confirmer la pertinence de l'industrialisation. Pour une meilleure appropriation, il faudra également s'assurer que le modèle d'intelligence artificiel s'intègre parfaitement dans le processus métier des utilisateurs finaux : interfaces, data visualisation, interprétabilité.

LES FREINS TECHNIQUES

Plusieurs difficultés techniques et questions peuvent émerger et se révéler être des points de douleur pour la mise en production d'un cas d'usage. En particulier, les algorithmes développés en phase de POC sont-ils scalables au volume de données en production et à la capacité de la plateforme à les traiter ? Sous quelle contrainte de temps ces algorithmes peuvent-ils être entraînés et délivrer les prédictions? Les données nécessaires seront-elles disponibles à temps pour pouvoir les traiter dans les délais imposés par des contraintes opérationnelles ?

En terme de code, l'exigence va se porter sur la modularité et la robustesse, via la mise en place de tests unitaires et test d'intégration (voir chapitre 3).

L'industrialisation va également nécessiter de mettre en place des pipelines de données faisant appel à des compétences complémentaires, hors de la palette des équipes de data scientists ayant participé à la phase de modélisation du POC.

La mise en place de nouveaux process, facilitant l'intégration d'améliorations futures ou de nouvelles fonctionnalités - provenant potentiellement d'autres équipes - sera également un point d'attention technique lors de la mise en production. C'est pourquoi la documentation technique et fonctionnelle du projet est capitale mais malheureusement souvent négligée à cette étape d'un projet afin d'assurer la fluidité et l'efficacité des interactions entre profils.

LE ROI ET L'INTÉGRATION DU CAS D'USAGE DANS LA STRATÉGIE

La phase de cadrage préliminaire peut avoir omis de vérifier certains points essentiels. Le manque de clarté sur le cas d'usage, son but global dans la stratégie d'entreprise. A ce titre, un frein majeur observé est l'absence de métrique sur le ROI du projet, et la vérification de ce ROI tout au long des phases projet. L'estimation en phase de cadrage d'un "*Profit & Loss*" permet de vérifier l'adéquation du projet avec la stratégie de l'entreprise.

QUESTIONS À CHRISTELLE LUGUET ET FRÉDERIC DE JAVEL (MAIF)

Christelle Luguet est chargée d'action développement des réseaux et Frédéric de Javel, chef de projet data sciences à la MAIF.
Suite à la mise en place du projet de qualification des mails, ils retiennent trois « keys learnings » nécessaires à la mise en place et à la réussite d'un projet IA.

Quelle démarche avez-vous adopté dans le cadre de ce projet ?

Le projet de qualification des mails a été initié de manière spontanée, l'idée était « On va essayer ». Il s'est démarqué au démarrage par une phase de test et d'exploration **sans à priori**. Cette phase de 3 mois en « lab », sous **un format « hors cadre en mode start-up »** a permis de jeter un regard nouveau sur la problématique et de s'inscrire dans une démarche pragmatique peu coûteuse en temps et en énergie.

Quel était le rôle de chacune des parties prenantes du projet ?

Les data scientists ne sont pas assureurs et inversement. Les data scientists connaissent bien le champ des possibles et les limites offertes par leurs algorithmes. A l'inverse ils n'ont que peu d'intuition sur les processus opérationnels. Chacun, data scientists comme assureur, doit faire un effort **pour parler simple sans simplifier**. Il faut ensemble comprendre les besoins, toucher les problématiques, les cas d'usages étudiés par des exemples. Il y a une immersion dans le quotidien d'un expert métier autre que le sien.

Il nous faut ainsi **apprendre les uns des autres de l'autre et partager une vision des gains opérationnels** attendus pour l'ensemble des acteurs.

Comment ce projet d'IA s'est-il intégré à l'écosystème MAIF ?

L'IA peut être source de craintes pour les collaborateurs : crainte de ne pas suivre la technologie, crainte que celle-ci fasse aussi bien qu'eux, crainte de perdre un jour leur emploi. Un des enjeux est d'accompagner l'ensemble des collaborateurs dans cette transformation. L'objectif de l'IA est de mettre les collaborateurs en pleine capacité, pour faire émerger, en toute confiance, une véritable intimité avec les clients. La mise en place de cette technologie lors du projet ne leur a rien enlevé mais au contraire apporté un confort de travail. A la MAIF, **l'objectif est d'utiliser l'IA pour accompagner la relation avec leurs sociétaires, augmenter le contact humain quand cela est nécessaire et proposer quand cela est possible de répondre à leurs besoins par une autre solution simple et efficiente.**

C. QU'EST CE QUE LA MISE EN PRODUCTION ET POURQUOI L'ANTICIPER EN DÉMARRAGE PROJET ?

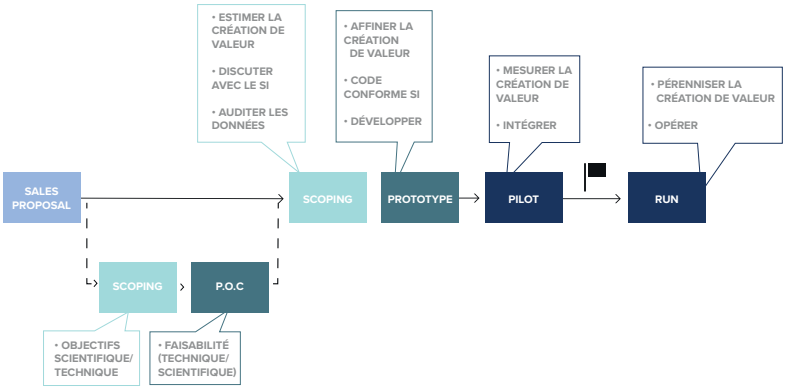
FACTEUR CLÉ DE SUCCÈS : ANTICIPER LA MISE EN PRODUCTION

Dès le début du projet, l'identification des phases, objectifs, livrables va permettre de projeter plus facilement le projet vers sa mise en production.

Chaque phase doit avoir des objectifs clairs et doit avoir son cadrage propre. Le passage d'une phase à l'autre doit s'effectuer en fonction de critères précis, avec des conditions de poursuite à la phase suivante (GO) ou d'arrêt du projet (NO GO). Des livrables sont également clairement définis pour chaque phase, et les rôles et responsabilités des acteurs sont définis pour chaque livrable.

Le coût d'entrée de cette méthode (cf figure ci-après) est plus élevé que sur l'approche classique (cadrage/POC/pilote/industrialisation) et sera particulièrement adapté chez des clients matures ayant déjà fait l'expérience du POC piégé dans un Datalab. Cependant, les différences entre les approches prototype et POC ainsi que leurs attendus associés doivent être présentées de manière pédagogique aux acteurs du projet.

Bien entendu, ce cadre n'est pas rigide et doit rester flexible par rapport aux méthodes de travail utilisées (agile, SAFe...) et les exigences doivent être travaillées tout au long du projet en les enrichissant ou en les modifiant au cours des phases.



Méthodologie Quantmetry de phasage d'un projet data anticipant l'industrialisation, attendus et livrables

ETAPES PROJET ET NIVEAUX DE MATURITÉ

Pour y voir plus clair, distinguons les phases projet en fonction de leur maturité et de leurs attentes.

Cadrage : il prépare le projet en terme de planning, ressources, charge et cadrage du besoin métier (cadrage simple). Si la phase suivante est un Prototype, le cadrage doit également contenir un recueil des contraintes techniques (cadrage approfondi).

POC : il permet de tester le potentiel réel d'une idée incomplète. Il ne s'agit pas de livrer l'idée, mais de démontrer si c'est faisable.

Prototype : un prototype simule le système complet ou au moins une partie importante de celui-ci et permet de montrer comment il sera réalisé. Un prototype doit fournir des composants réutilisables dans une version pilote ou de production.

Pilote : un pilote est un système en production disponible pour un sous-ensemble du périmètre. La raison d'être du pilote est de mieux comprendre comment le système complet sera utilisé sur le terrain et de l'affiner.

Run : le run représente le déploiement de la solution finale sur un périmètre global et avec une automatisation complète. La solution est alors dans les mains des utilisateurs finaux.

LES CRITÈRES DE GO/NO GO À CHAQUE ÉTAPE DU PROJET

La fin de chaque étape marque l'aboutissement d'une partie du projet qui a nécessité un investissement financier et humain. C'est l'heure de faire un bilan sur les réussites et difficultés rencontrées, mais aussi de mettre en place un processus de vérification des conditions favorables à la poursuite du projet.
La phase de cadrage est en effet conditionnée au niveau d'alignement avec la stratégie, à la maturité et aux risques du cas d'usage sélectionné, à la valeur business mais aussi à l'estimation de la rentabilité du projet.

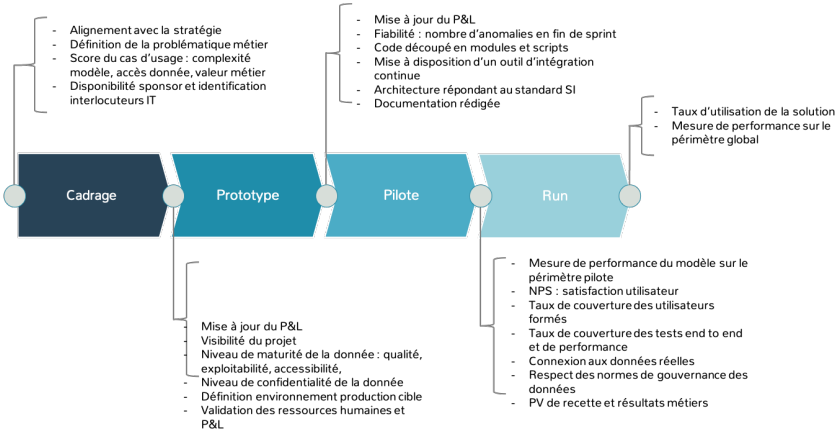
A la fin de cette phase, l'objectif métier du cas d'usage est clairement défini et la maturité des données a été évaluée. Ce scoring de maturité des données peut mener à l'arrêt du projet, si les données ne sont pas d'un niveau de

qualité, d'accessibilité ou d'exploitabilité suffisant. Ensuite, il faudra avoir identifié clairement les sponsors et valider leur disponibilité, puis définir avec l'IT la plateforme cible et l'équipe projet nécessaire.

A l'issue de la phase de prototype, la mise à jour du P&L confirme l'estimation faite à la phase précédente, et le code bien architecturé, documenté est livré. Enfin, la mise à disposition d'un outil d'intégration continue et l'intégration de l'architecture cible dans le SI de l'entreprise sont une condition *sine qua non* de poursuite du projet en phase pilote.

Le pilote sera évalué sur plusieurs critères reposant sur la mesure de performance du modèle sur des données réelles, bien entendu, mais également sur les taux de couverture des tests end-to-end, le respect des normes de gouvernance de la donnée assurant la robustesse du système en production. Autre point important, la satisfaction du métier, et des utilisateurs de la phase pilote est un point essentiel à évaluer afin d'envisager la phase de run qui suivra.

Le succès et la pérennité de la phase de run sera mesurée par l'utilisation réelle de la solution dans le processus métier et par la satisfaction des utilisateurs, et par la performance du modèle étendu au périmètre global.



LE POC, PASSAGE OBLIGÉ ?

Les preuves de concept (POC) constituaient il y a quelques années les premiers

projets des organisations data naissantes. Depuis, les typologies de projet ont évolué en même temps que la maturité des entreprises. Nous avons pu observer depuis une modification du mandat du POC. Là où l'objectif initial était de démontrer la faisabilité et pertinence d'un concept ou d'une approche, nous nous retrouvons à développer des prototypes sous l'appellation de POC. Cette transformation est importante car elle change le mandat et l'attente liés à cette phase initiale du projet. Un POC, dans son essence, n'a pas vocation à être industrialisé et on est souvent bien loin d'un outil/produit lorsque celui ci touche à sa fin contrairement à un prototype, qui vise une industrialisation conditionnée à un ROI. Si la question du ROI ne peut pas être écartée lors de la préparation d'une industrialisation, il en va autrement du POC. Ce dernier n'est justifié que si le projet repose sur des concepts encore non démontrés ou éprouvés.

Ainsi, lorsque l'on sait que le projet va aller en production car le concept est déjà bien éprouvé dans son secteur industriel (par exemple, un *retailer* souhaite réaliser des prévisions de vente, une banque cherche à industrialiser un modèle de *churn*), alors il convient d'intégrer dès les départ du projet une vision sur l'architecture cible et les contraintes associées au projet. On change alors de paradigme, d'une vision exploratoire à une vision produit, pour accélérer la mise en production.

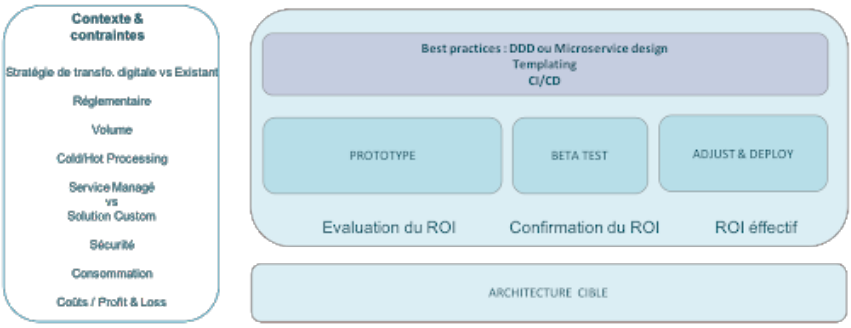
	Offre P.O.C	Offre Prototype
Coûts	+	++
Objectif	Gains scientifique ou technique	Gains métier - financier
Ressources techniques	Locales / expérimentales	Environnement Datalab / SI
Audience cible	Interne Datalab	Utilisateurs « early adopters »
Réutilisation	Faible	Elevée
Interaction utilisateurs	Faible	Première version de l'interface utilisateur
Principaux livrables	✓ Statistiques descriptives ✓ Notebooks d'exploration et modélisation des données ✓ Tests unitaires ✓ Document scientifique décrivant la démarche et les choix ✓ P&L estimé	✓ Code modulaire ✓ Contrat d'interface ✓ Documentation d'architecture générale ✓ Configuration d'environnement ✓ Document scientifique décrivant la démarche et les choix ✓ P&L consolidé
Quand l'adopter?	Le P.O.C est nécessaire lorsque l'idée n'a pas encore été acceptée, que vous souhaitez tester une hypothèse ou une idée innovante. Cela vous	Le prototype est une méthode utilisée pour visualiser et comprendre comment le système fonctionnera et comment il sera reçu par

POC ou prototype : quelle stratégie à adopter?

CHANGER DE VISION : DE L'EXPLORATION À LA VISION PRODUIT

Déployer un code en production c'est mettre à disposition un produit à des utilisateurs. Avant de construire le produit, avant même de concevoir le produit, il est important d'identifier ses utilisateurs : à qui ce nouveau produit est-il destiné ? Quelques questions peuvent suffir à impacter fortement le projet dans son ensemble.

QUESTIONS LIÉES AU UTILISATEURS	EXEMPLE D'IMPACT STRUCTURANT SUR LE PROJET
A QUELLE DIRECTION APPARTIENNENT LES UTILISATEURS ?	Qui va payer pour le projet ?
COMBIEN D'UTILISATEURS ?	Dimensionnement de l'infrastructure de production. Dans le cas de ressources d'infrastructure limitées: dimensionnement de la complexité des modèles qui pourront être utilisés
COMMENT LES UTILISATEURS ACCÈDENT-ILS À L'INFORMATION ?	Les possibilités suivantes ont un impact complètement différent en terme de développement • Un nouveau site web • Intégré à leur application métier existante • Les résultats sont envoyés par mail à tous les utilisateurs
OÙ SONT LES UTILISATEURS ?	En déplacement ? Dans un site de l'entreprise à l'étranger ? Selon la situation la mise à disposition de produit sera plus ou moins complexe en terme de sécurité réseau.
QUELLE RAPIDITÉ DE MISE À JOUR EST NÉCESSAIRE POUR QUE L'UTILISATEUR PUISSE PRENDRE UNE DÉCISION DANS SON PROCESSUS ? Par exemple: • immédiat si j'ai besoin d'une information sur un client qui vient de me contacter par téléphone • tous les mois si je dois faire des planifications budgétaires	Quels types de modèles d'apprentissage artificiels sont à ma disposition ? Online-learning vs batch-learning ?



Prise en compte du contexte et contraintes dans un modèle orienté produit pour accélérer la phase de mise en production

D. QUELS SONT LES JALONS CLÉS APRÈS LA MISE EN PRODUCTION DU MODÈLE ?

La mise en production d'un modèle n'est que le début d'une phase opérationnelle qui va nécessiter une mise en place de contrôles sur :

- L'ingestion des données, accompagnée de KPIs à mettre en oeuvre, ainsi que sur la QA/QC (Quality Assurance/Quality Control)
- le contrôle opérationnel du run, sur les données d'entrée, sur les performances (comparaison entre les performances théoriques et celles obtenues), sans oublier la vérification des biais (par exemple : mon algorithme des scoring de crédit est-il éthique, n'intègre-t-il pas de biais sociétaux ?)

De plus, elle va nécessiter de nouveaux process dans l'optique de poursuivre le cycle itératif du développement du produit et de le faire évoluer, via la mise en place d'une démarche CI/CD (Continuous Integration/Continuous Delivery), d'A/B testing de modèles et la définition de la stratégie de remplacement d'un modèle en production.

RETOUR D'EXPÉRIENCE

Nous pouvons citer comme exemple l'une de nos missions réalisée dans le domaine de l'assurance où nous avons mis en production des modèles de churn pour divers contrats, basés sur un socle de données commun. Les bonnes performances de la phase de prototype n'ont pas été retrouvées en production pour l'ensemble des contrats, mais uniquement pour une sous-partie d'entre-eux. Il s'agit là d'un exemple où la **dérive des données**, assez difficile à objectiver, n'a pas le même impact sur les différents scores construits. Ce constat a pu être fait sur plusieurs de nos missions, ce qui milite pour une approche de *feature sélection plus rigoureuse que les pratiques usuelles de data science, où le filtrage du signal depuis les données est laissé au modèle, le data scientist tentant alors de construire le plus de features signifiantes pour lui... mais pas forcément les plus pertinentes d'un point de vue métier.*

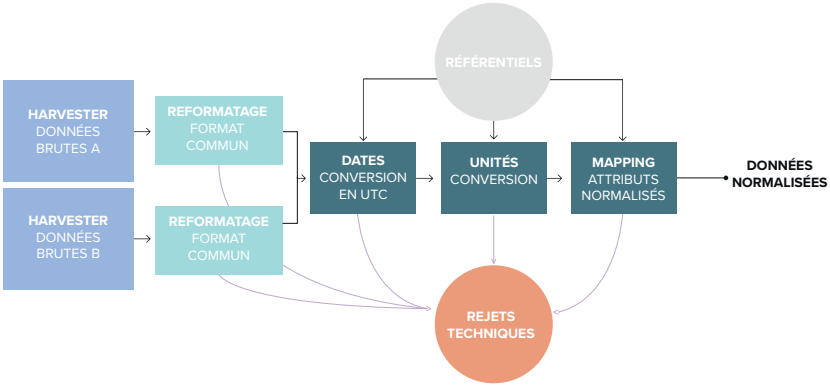
E. L'IMPORTANCE CRUCIALE DE LA "PLOMBERIE DATA", OU COMMENT GÉRER LES DONNÉES EN PRODUCTION ?

En phase de production, l'alimentation du modèle en données sera assurée par un pipeline d'extraction, transformation et chargement (ETL), et la gestion de la qualité de ces données, accompagnées de la mise en place d'indicateurs de performances.

Prenons l'exemple d'un pipeline de traitements de données suivant :

- des données brutes proviennent de capteurs de différentes sources (constructeurs différents)
- les données brutes subissent des traitements pour mettre en forme les données en entrée, convertir les dates en UTC, convertir les unités et appliquer un mapping sur les tags en entrée, afin d'être indépendants des sources.
- les données normalisées sont alors traitées par un process de QA/QC afin de les nettoyer pour les mettre à disposition des utilisateurs

Cela se traduit par ce schéma :



À chaque étape du pipeline, des rejets techniques sont opérés :

- Le format des données brutes reçues n'est pas celui attendu
- La référence du capteur présent dans les données brutes est inconnu du référentiel
- La conversion de la date des données brutes vers une date en UTC échoue (date au mauvais format, problème de conversion)
- La conversion (vers une unité définie) de la valeur du capteur est impossible (soucis dans le typage de la valeur, unité de la source inconnue, formule de conversion non précisée)
- L'application d'un mapping entre le tag provenant de la donnée brute et un attribut "officiel" (exemple : Temp pour le constructeur A, Temperature pour le constructeur B en Tmp -nom officiel) échoue car le tag est inconnu du référentiel

Ces rejets techniques doivent faire l'objet de KPI permettant :

- d'apprécier leur évolution dans le temps
- d'agir en conséquence (modification du paramétrage, remonter un incident à la source etc.)

Ces KPI s'accompagnent de méta-données pertinentes afin de produire des axes d'analyse :

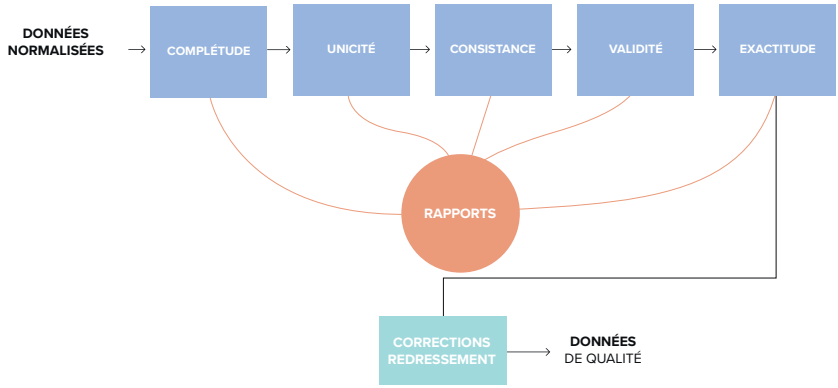
- la date (granularité à définir en fonction du besoin)
- le constructeur du capteur
- le type d'équipement
- etc.

L'ajout de ces méta-données permet de mettre en évidence un éventuel problème suite à une mise à jour opérée par un constructeur par exemple.

Le stockage des erreurs techniques dans un outil comme ElasticSearch est approprié :

- rejetées
- visualiser les rejets sous forme graphique, filtrer, zoomer et analyser

Une fois ces étapes techniques franchies, les données "non rejetées" sont "normalisées".
Les données normalisées peuvent alors être traitées par le traitement de QA/QC, afin de vérifier fonctionnellement la qualité des données.



Les rejets fonctionnels peuvent par exemple être générés à partir des contrôles suivants :

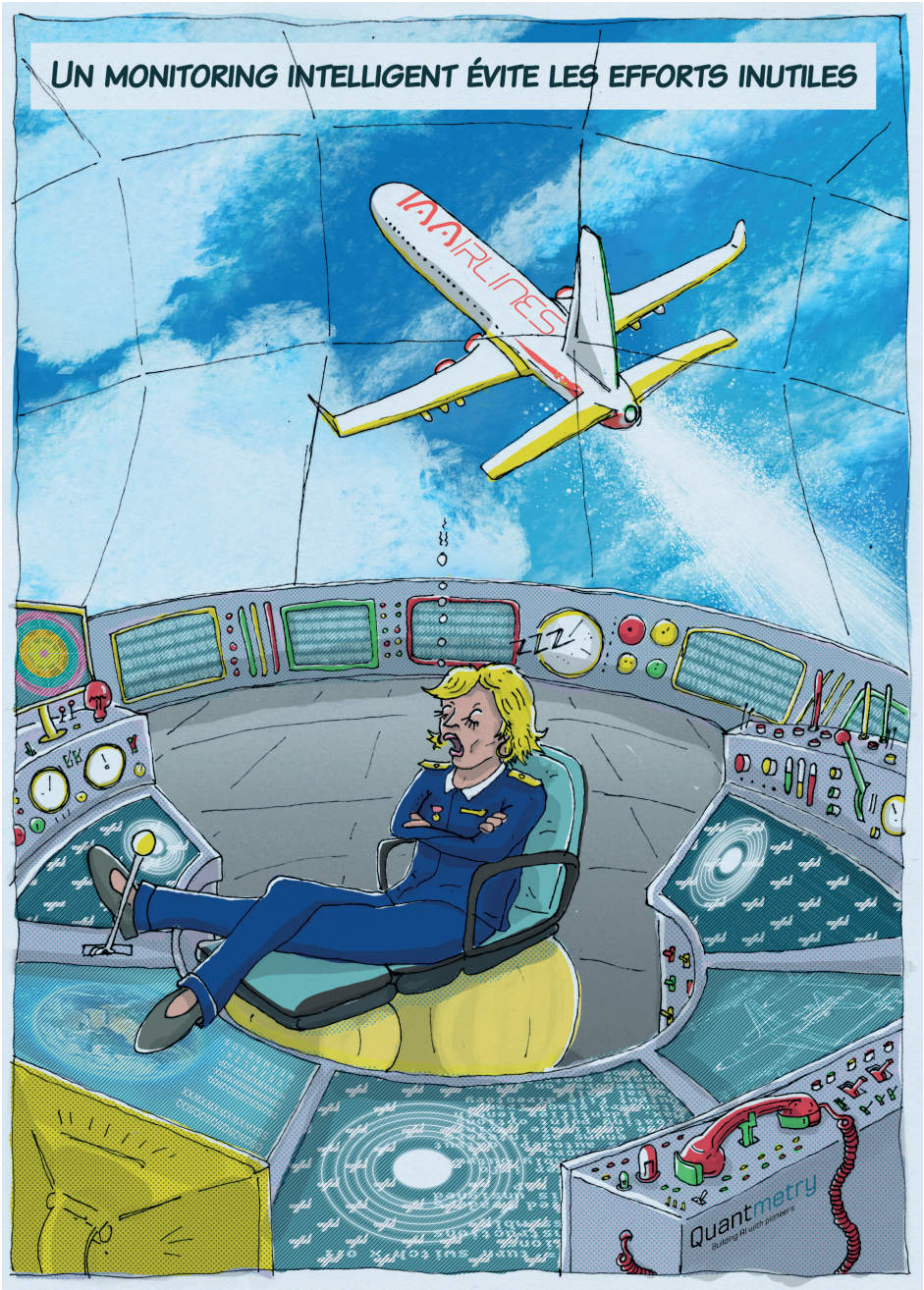
- Complétude (a-t-on le nombre de valeurs attendu ?)
- Unicité (a-t-on des valeurs uniques ?)
- Consistance (les valeurs évoluent-elles dans le temps ?)
- Validité (les données sont-elles comprises dans une plage acceptable ?)
- Exactitude (les données sont-elles cohérentes entre elles ?)

Tout comme les rejets techniques, les rejets fonctionnels doivent être suivis par des KPI, avec un maximum d'axes d'analyses pour comprendre au mieux l'origine des dysfonctionnements.

En s'appuyant sur les rapports générés, les données sont alors corrigées :

- suppression des données en doublon en ne conservant que la valeur la plus récente par exemple
- ajout de valeurs manquantes ou aberrantes par interpolation
- etc.

Les données corrigées / redressées sont alors mises à disposition des utilisateurs.


UN MONITORING INTELLIGENT ÉVITE LES EFFORTS INUTILES
AIRLINES
Quantmetry
Building AI with pioneers

2. COMMENT MONITORER LES MODÈLES EN PRODUCTION ?

***Contributeurs** : Rémi Adon, Luis Blanche, Guillaume Mocquet*

Le cycle de vie des modèles est, dans une certaine mesure, la suite fonctionnelle du cycle de vie des données : si les données ont une vie, les modèles, *a fortiori*, aussi.

Après plusieurs années à monitorer et fiabiliser nos données, ce sont donc les mêmes motivations qui nous poussent aujourd'hui à monitorer nos modèles au sein des environnements de production.
Mais est-ce vraiment aussi simple ? Une différence importante subsiste entre le modèle entraîné et les données sur lesquelles il s'entraîne : un modèle d'apprentissage automatique n'est, malgré la profusion de publications récentes à ce sujet, pas uniquement consommateur de données, mais aussi **producteur de données.**

L'étude des modèles portera donc, en majorité, sur leur **comportement**, c'est-à-dire tout aussi bien sur les entrées reçues que sur les variations dans les sorties produites.

A. QUELS TYPES DE PROCÉDURES METTRE EN PLACE ?

DATA QUALITY CHECKS

S'assurer de la qualité des données en entrée est un point de passage nécessaire vers des prédictions de qualité ; dans la langue de Faulkner : *shit in, shit out*.

Les données en entrée sont donc en général soumises à un pipeline de validation, pouvant comprendre :

- complétude des données, vérification des valeurs manquantes ;
- cohérence des données, vérification de la validité des valeurs en fonction de règles métier ;
- format des données, vérification du respect de certaines interfaces à la réception.

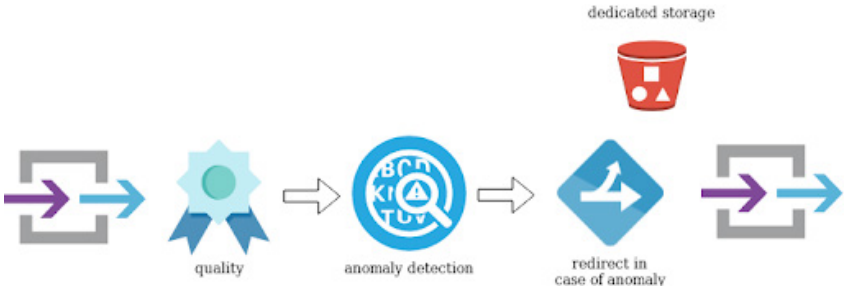
SYSTÈMES D'ALERTES ET REJETS FONCTIONNELS

Une question qu'il pourrait être légitime de poser est celle de la destination des données qui ne respectent pas les Data Quality Checks. Dans un projet de grande ampleur, il est en général conseillé de rediriger ces données malformées vers une destination dédiée, plutôt que de les corriger instantanément. Cette approche présente plusieurs avantages :

- Une complexité algorithmique en général moins élevée, les opérations de vérification pures étant en général moins coûteuses que les opérations d'imputation/correction de valeurs ;
- Un renversement dans la méthodologie du projet data : les investigations à venir porteront sur l'origine du problème (quelle étape dans le pipeline est responsable de ces malformations ?), plutôt que sur la mise à jour d'algorithmes de corrections pouvant rapidement devenir obsolètes ;
- Un périmètre (visualisation, stockage, etc..) dédié aux données malformées, et donc une agilité accrue sur ces problématiques.

Nous notons néanmoins que dans certaines situations, la correction automatique des données d'entrées est possible, l'imputation de valeurs manquantes ou aberrantes étant un champ d'études plutôt bien connu en apprentissage automatique.

Ci-dessous un exemple typique de pipeline



MODEL AS A SERVICE

Si intelligent soit-il, un modèle d'apprentissage automatique n'est rien d'autre, du point de vue fonctionnel, qu'un service. C'est à ce titre qu'il deviendra alors, sur une architecture *Cloud* par exemple, un citoyen de première importance, qui se doit de tenir certaines promesses pour être mieux compris et respecté par ses pairs.

Parmi ces promesses à tenir, certaines, comme le niveau de service[1], sont issues des pratiques communes en matière de déploiement logiciel. D'autres, en revanche, comme le suivi du *Time To Predict*, sont spécifiques à l'application de la philosophie micro-service au *Machine Learning*.

2 parties distinctes :

- ***Back-end :*** en charge d'entraîner (ou de ré-entraîner) le modèle. Les calculs sont complexes et longs et sont généralement réalisés sur une plateforme de *Cloud Computing*.
- ***Endpoint*** : en charge de réaliser les prédictions. De plus en plus, celles-ci sont réalisées localement, on parle alors de *Edge Computing*. Cela offre deux principaux avantages : les données n'ont pas besoin d'être envoyées à

1. *Service Level Agreement*

un opérateur tiers et le temps de réponse est optimisé, sans aucun échange réseau. C'est particulièrement appréciable lorsque la couverture réseau est limitée ou inexistante.

SUIVI DES DÉPENDANCES

Les modèles de *Machine Learning* consomment des données pouvant provenir de différents systèmes d'informations (database, APIs, etc.). Il est important de suivre les évolutions de ces systèmes pour s'assurer qu'elles n'impactent ni la donnée qui est fournie au modèle, ni la façon d'y accéder.

B. QUELLES TYPOLOGIES D'INDICATEURS SUIVRE ?

TAUX DE DISPONIBILITÉ / D'INDISPONIBILITÉ

Lorsqu'un modèle est en production, dès lors que les résultats de ce dernier sont utilisés par des applications opérationnelles critiques, il est important de suivre le taux de disponibilité / d'indisponibilité. Par exemple, dans le cas de l'optimisation de l'approvisionnement des stocks des magasins d'un grand groupe, il est important de suivre le taux de disponibilité de l'application en charge des prédictions pour déterminer le SLA global de l'application.

TEMPS DE RÉPONSE

Lorsqu'un modèle est en production, il doit souvent satisfaire à des exigences sur le temps de réponse aux requêtes (souvent défini dans les SLA). C'est donc l'un des premiers types d'indicateurs qu'il peut être intéressant de suivre. On citera par exemple :

- Le nombre de requêtes (si le niveau de requêtes chute, il y a peut-être un problème)
  - Exemple : suivre le niveau de requêtes et utiliser un modèle prédictif simple pour comparer le niveau réel au niveau attendu, remonter une alerte si la différence dépasse un certain seuil ;
- Le temps de réponse aux requêtes
  - Exemple : comparer avec un test statistique les distributions de temps de réponse normaux vs réels.

NOMBRE DE HITS / PRÉDICTIONS / CONNEXIONS SIMULTANÉES

Pour ne pas générer de goulot d'étranglement il est important de suivre le nombre de hits / prédictions / connexions que le modèle est capable de gérer en simultané sans entraîner de dégradation des performances.

PERFORMANCES PROPRES AU MODÈLE

Lors des étapes de construction du modèle, certains indicateurs faisant état de la performance du modèle sont suivis. Dans le cas de modèles d'apprentissage supervisé, ces indicateurs sont calculés lors de l'entraînement en comparant les labels et valeurs prédites avec celles du jeu d'entraînement et celles des jeux de validation et de test, qui ont été préalablement mis de côté pour vérifier le pouvoir de généralisation du modèle.

Pour un classifieur par exemple on calcule des métriques comme la précision, le rappel ou encore l'aire sous la courbe ROC ou PR.

Il convient alors, dans une phase de run, de s'assurer que ces indicateurs ont droit à un suivi régulier, dans le but de déceler, ne serait-ce que visuellement, une anomalie dans la performance du modèle.

Lorsque qu'un modèle est en production et fournit des prédictions en fonction des requêtes qu'il reçoit, on ne dispose pas a priori des vraies valeurs, puisque c'est le rôle du modèle de les prédire. Pour calculer ces indicateurs, il faudra donc nécessairement labelliser les données fournies en entrée par ces requêtes *a posteriori*. Dans le cadre d'un forecast (prédiction de séries temporelles), il suffit d'attendre que les événements se réalisent pour les comparer avec les prévisions, mais dans un contexte de classification d'images par exemple, il sera nécessaire que des humains labellisent un échantillon de données pour évaluer la qualité des prédictions.

Cette labellisation peut donc être coûteuse et il faut bien évaluer au préalable la fréquence et le volume à adresser pour s'assurer de la constance des métriques.

Il est envisageable de mettre en place des solutions de type interface utilisateur ou même d'acheter des prestations d'annotations pour accélérer les annotations sur les prédictions récentes et mieux sécuriser la validité du modèle au cours du temps.

PROBLÈME DU DÉLAI D'OBTENTION DE LA "*GROUND TRUTH*"

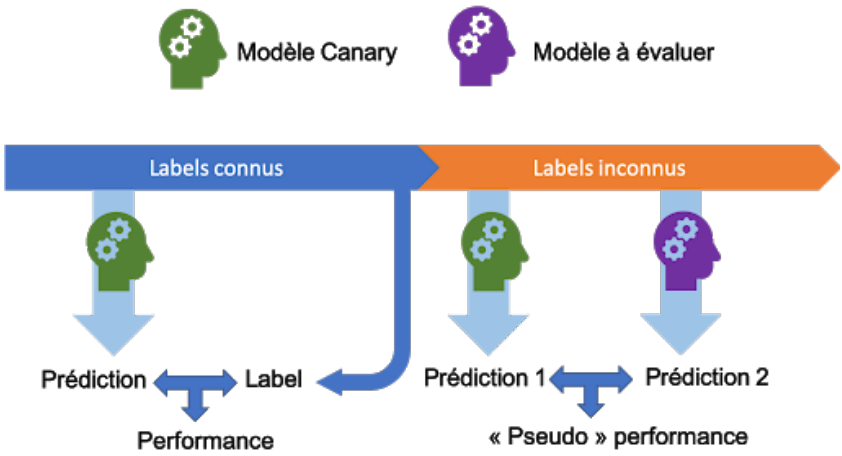
Selon les tâches de prédiction traitées, il y a parfois un délai dans l'obtention des données réelles qui est supérieur au délai nécessaire au monitoring du modèle. Ainsi, il est parfois nécessaire de suivre des métriques qui sont différentes d'une comparaison entre les labels prédits et les labels réels (puisqu'ils ne sont pas disponible au moment du monitoring).

De nombreux modèles supervisés ont pour sortie une probabilité (on passe à une classe à l'aide d'un seuil dans le cas de la classification). Il peut être intéressant de suivre la distribution de ces probabilités et vérifier par un test statistique (exemple Kolmogorov-Smirnov, tests d'appariement en général, test des rangs signés de Wilcoxon-Mann-Whitney) qu'elle ne diffère pas de la distribution historique.

Ceci étant dit, il est possible d'avoir des variations de distribution qui ne correspondent pas à des erreurs mais à des situations bien réelles. Prenons un exemple de système de détection d'un chat dans un image. Lors de l'entraînement on a pu observer que 10% des images contenaient des chats. Un suivi de cette proportion à la prédiction, habituellement autour de cette proportion, peut augmenter et atteindre 50%. Ceci est un motif d'alerte, car inhabituel, mais peut résulter d'une nouvelle tendance sur les réseaux sociaux.

CANARY MODEL

On peut utiliser un modèle plus ancien, validé à l'aide de la *ground truth* obtenue après le délai cité plus haut pour comparer les prédictions du modèle actuel à celle de ce modèle, appelé modèle *canary*. On fait l'hypothèse que les prédictions de ce modèle sont assimilables à la *ground truth* et on les utilise comme labels dans les calculs des métriques de performance.



DÉVIATION ENTRAÎNEMENT / SERVING

Il n'est pas inhabituel d'avoir des bases de codes différentes pour les données d'entraînement et les données de prédiction. Les contraintes de prédiction sur le temps de réponse notamment font que certaines variables de prédiction peuvent être calculées de façon différentes entre l'entraînement et la prédiction (par exemple code d'entraînement en Python et prédiction en C), et il faut s'assurer qu'elles prennent exactement les mêmes valeurs dans les deux cas. En pratique il faut essayer de mesurer la déviation entre les données de prédiction et les données d'entraînement, cela peut se faire en récupérant des échantillons des données de prédiction ainsi qu'à l'aide à nouveau de tests statistiques sur les différentes variables ou de simples statistiques (maximum, moyenne, données manquantes, ...) associées à des seuils d'alerte.

EVOLUTION DE L'INTELLIGIBILITÉ

Depuis quelques années, l'intelligibilité en apprentissage automatique[2] est un domaine de recherche à part entière, qui trouve de plus en plus d'applications concrètes. Il est vital de garantir un niveau de transparence élevé pour que les solutions soient effectivement adoptées par les clients. Cette thématique faisait d'ailleurs l'objet de notre livre blanc 2018, "IA explique toi".

Concrètement il s'agit de proposer des liens entre les variables explicatives et les prédictions effectuées. Des techniques récentes comme SHAP (1) permettent

2. *explainable Machine Learning*

d'identifier le poids de chaque variable dans la prédiction d'un unique individu. Ce sont des données qu'il peut être intéressant de monitorer, et dont on peut essayer de vérifier qu'elles ne dérivent pas.

En effet, elles peuvent nous permettre de repérer une erreur dans une variable explicative calculée au moment de la prédiction, qui aurait par exemple pour conséquence qu'elle prendrait beaucoup plus d'importance dans la prédiction.

C. LES OUTILS EXISTANTS

La majorité des outils utilisés sur des projets data seront en fait des outils de DevOps, *Data Engineering*, utiles dans les procédures vues précédemment (rejets fonctionnels par exemple), mais aussi des outils de visualisation de données qui seront, eux, privilégiés dans une optique de suivi des indicateurs sur le modèle.

EXEMPLE D'OUTILS CHEZ LES CLOUD PROVIDERS

Les principaux services des grands fournisseurs de solutions Cloud tels qu'Amazon, Google ou Microsoft intègrent nativement des outils de monitoring et de gestion des logs. Il est possible de réaliser des actions sur réception des triggers issus de ces outils.

Microsoft Azure permet de collecter des données d'un modèle en production :

- les données d'entrée du modèle pour les web services déployés sur Azure Kubernetes Cluster (2)
- les prédictions associées

Il est donc possible de déposer ces données sur un stockage blob au fur et à mesure de leur production ce qui permet de construire des visualisations de suivi via Microsoft Power BI par exemple.

Google Cloud fournit un service équivalent avec son AI platform (3) qui est configurable pour permettre le logging des données de prédiction (input + prédictions) dans StackDriver (4).

Concernant Amazon Web Services, voir l'interview d'Olivier Cruchant, Machine Learning Solutions Architect, en fin de chapitre.

PROMETHEUS

Prometheus (5) est une solution open source bâtie par SoundCloud. Elle propose une boîte à outils de suivi de métriques et d'alertes (plutôt opérationnelles). Le paradigme adopté est celui des séries temporelles et permet donc facilement de faire du *forecast* et de la détection d'anomalies pour repérer en avance d'éventuelles dérives. C'est une solution qui a été adoptée dans beaucoup de cas où on a besoin de détecter et de réagir très rapidement aux problèmes pouvant se présenter lorsqu'un modèle est déployé et mis à disposition de grandes populations d'utilisateurs.

SOLUTION DE MONITORING "SUR ÉTAGÈRE"

Il y a aujourd'hui peu de solutions de monitoring spécifiques aux modèles d'apprentissage automatique sur étagère. Il existe une diversité énorme dans les environnements, les exigences et les types de modèles que l'on veut monitorer. Ainsi il y a de nombreuses façons de présenter le suivi et la solution privilégiée aujourd'hui est de construire des tableaux de visualisation ad hoc pour pouvoir s'adapter aux particularités de chaque modèle.

Un exemple d'une telle solution sur étagère est Sonar d'Hydrosphère (6), qui permet d'identifier les dérives, de détecter des anomalies, de gérer les ré-entraînements et de remonter des alertes.

GESTION DES LOGS

La brique fondamentale pour le monitoring est la gestion des logs, qui peuvent parfois s'accumuler en grande quantité. Ainsi il est souvent utile de disposer d'un outil d'exploration permettant de gagner du temps. Les services de gestion des logs des cloud providers ont été cités plus haut.
Datadog (6) est indépendant et permet de monitorer des applications et de stocker des logs quelque soit l'endroit où elles sont déployées.
Lorsque les logs sont trop massifs pour être analysés à l'oeil nu, il peut être intéressant de mettre en place un stack ElasticSearch-Logstash-Kibana (ELK) qui permettra de requêter et visualiser les logs de façon efficace (ex: Logz.io (7)).
Splunk (8) est également une solution populaire qui embarque notamment des algorithmes pour faciliter l'exploration et l'exploitation des logs machine.

OUTILS D'ANNOTATION

Comme cité précédemment, il est capital de disposer de vrais labels pour évaluer les prédictions effectuées. Lorsqu'il est impossible de les obtenir automatiquement, il faudra passer par une labellisation humaine. Les différents cloud providers proposent des services de labellisation. Si vous disposez des moyens humains pour le faire en interne, certains outils comme Prodigy (9) ou Supervisely (10) permettent d'accélérer et de faciliter grandement l'annotation.

Liste de lecture :

› *Instrumentation, Observability and Monitoring of Machine Learning Models.* Conférence de Josh Wills de Slack. (https://www.infoq.com/presentations/instrumentation-observability-monitoring-ml/)

› *The ML Test Score: A Rubric for ML Production Readiness And Technical Debt Reduction.* Papier de Google (https://ai.google/research/pubs/pub46555)

› *AWS Well Architect Framework. Papier de AWS* (https://d1.awsstatic.com/whitepapers/architecture/AWS_Well-Architected_Framework.pdf)

QUESTIONS À OLIVIER CRUCHANT (AWS)

Olivier Cruchant est Specialist Solutions Architect ML chez Amazon Web Services. Il accompagne les entreprises dans l'utilisation des services spécialisés d'AWS pour leurs projets de Machine Learning et Deep Learning.

Quels sont les outils d'AWS qui permettent d'accompagner le cycle de vie des modèles ?

AWS propose une suite d'outils pour accompagner l'ensemble du cycle de vie des modèles.

Amazon S3 est adapté pour le stockage des données d'entraînement. Le service permet de stocker différents formats de données : images, vidéos, texte ou des formats en colonne comme Parquet. Il est scalable sans limite et est doté d'une tarification économique. Ces trois points permettent une itération rapide à bas coût sur les données, particulièrement utile en Machine Learning (ML) où les premières itérations sont généralement exploratoires et au ROI incertain.

Pour ce qui est de la transformation des données, AWS propose différents services offrant des niveaux d'abstraction et d'expression complémentaires. Par exemple, Amazon Athena permet de requêter de manière interactive de gros volumes de données stockés sur Amazon S3 via une interface SQL, sans avoir à provisionner de serveur. De même, AWS Glue permet de créer un catalogue de métadonnées et de développer des tâches Spark ou Python entièrement gérées, c'est-à-dire sans avoir à configurer son propre cluster de calcul. Un niveau de flexibilité plus grand est possible via Amazon Elastic Map Reduce (EMR), qui permet de créer des clusters de calcul dotés des dépendances de son choix (Spark, Presto, Hive, HBase, TensorFlow entre autres). Enfin, Amazon Redshift est un service d'entrepôt de données qui permet de requêter à très faible latence via SQL des tables de données structurées.

Amazon SageMaker est un service modulaire offrant des fonctionnalités qui couvrent tout le cycle de vie d'un modèle de ML, de l'exploration au déploiement. SageMaker Ground Truth permet d'annoter des données par *crowdsourcing* ou via des équipes privées administrées par les clients, et contribue à une réduction des coûts d'annotation via l'usage optionnel de l'apprentissage actif qui permet de confier une portion de l'annotation à des machines. Amazon SageMaker

permet également de développer, entraîner et déployer ses propres modèles dans le framework de son choix, et des environnements Docker sont disponibles sur étagère pour TensorFlow, Scikit-Learn, PyTorch et Apache MXNet. La recherche de réglages haut-niveau d'un modèle (hyperparamètres) est une étape traditionnellement longue et coûteuse car généralement effectuée par force brute. Amazon SageMaker propose de remplacer cette recherche par force brute par un modèle d'optimisation bayésienne, qui permet à la fois de réduire les coûts et d'accélérer l'itération. En ce qui concerne l'optimisation du déploiement, Amazon SageMaker permet d'accélérer l'inférence de manière matérielle - en utilisant des accélérateurs tels que les GPUs NVIDIA ou Amazon Elastic Inference - et de manière logicielle, en compilant les modèles vers une représentation et un environnement optimisé pour chaque plateforme, via le compilateur SageMaker Neo, dont l'environnement d'exécution a été publié en open-source.

Amazon SageMaker stocke les métadonnées associées aux modèles et persiste leurs artefacts, ce qui facilite la navigation dans les expériences et la traçabilité.

Concernant l'instrumentation, la mesure des métriques du modèle et de l'infrastructure associée (GPU, CPU, mémoire, disque) est disponible dans Amazon CloudWatch pour les tâches d'entraînement comme pour les tâches d'inférence. Amazon SageMaker permet de déployer plusieurs modèles partageant les mêmes variables d'entrée derrière un seul *endpoint*, afin de comparer leurs métriques et d'effectuer des déploiements avec recouvrement.

Concernant la persistance de version du code scientifique et du code d'orchestration, les instances de développement et le SDK Python permettent une connexion aux répertoires git. Une bonne pratique de développement est de séparer le code scientifique du code d'orchestration, un comportement natif dans Amazon SageMaker, où le code scientifique est exécuté dans un container docker.

Quelles sont les principales considérations concernant le réentraînement des modèles ?

Si on exclut les réentraînements consécutifs à des ajustements proactifs du système (nouvelle architecture, ajout de variable d'entrée) les considérations sont les suivantes :

Tout d'abord, si le modèle est constitué de sous-systèmes, se demander s'il faut désynchroniser le réentraînement de chaque sous-système. Par exemple :

- Sur un réseau neuronal, avoir des fréquences d'entraînement différentes pour

différentes sections du graphe.

- Sur un enchaînement de modèles indépendants, par exemple un détecteur généraliste d'objets suivi d'un classificateur spécialisé, les différents modèles peuvent être réentraînés à des fréquences différentes.

Ensuite, les facteurs de déclenchement du réentraînement des différents modèles seuls seront :

- *Est-ce qu'il y a eu un changement important dans l'input ?* Ce changement peut se détecter en surveillant la distribution de l'input et, lorsque c'est possible, en connaissant son environnement et en anticipant les changements des systèmes amonts. Par exemple, comment anticiper ou détecter si un système amont qui communique une variable de prix au modèle passe d'une représentation euro à une représentation dollar ?

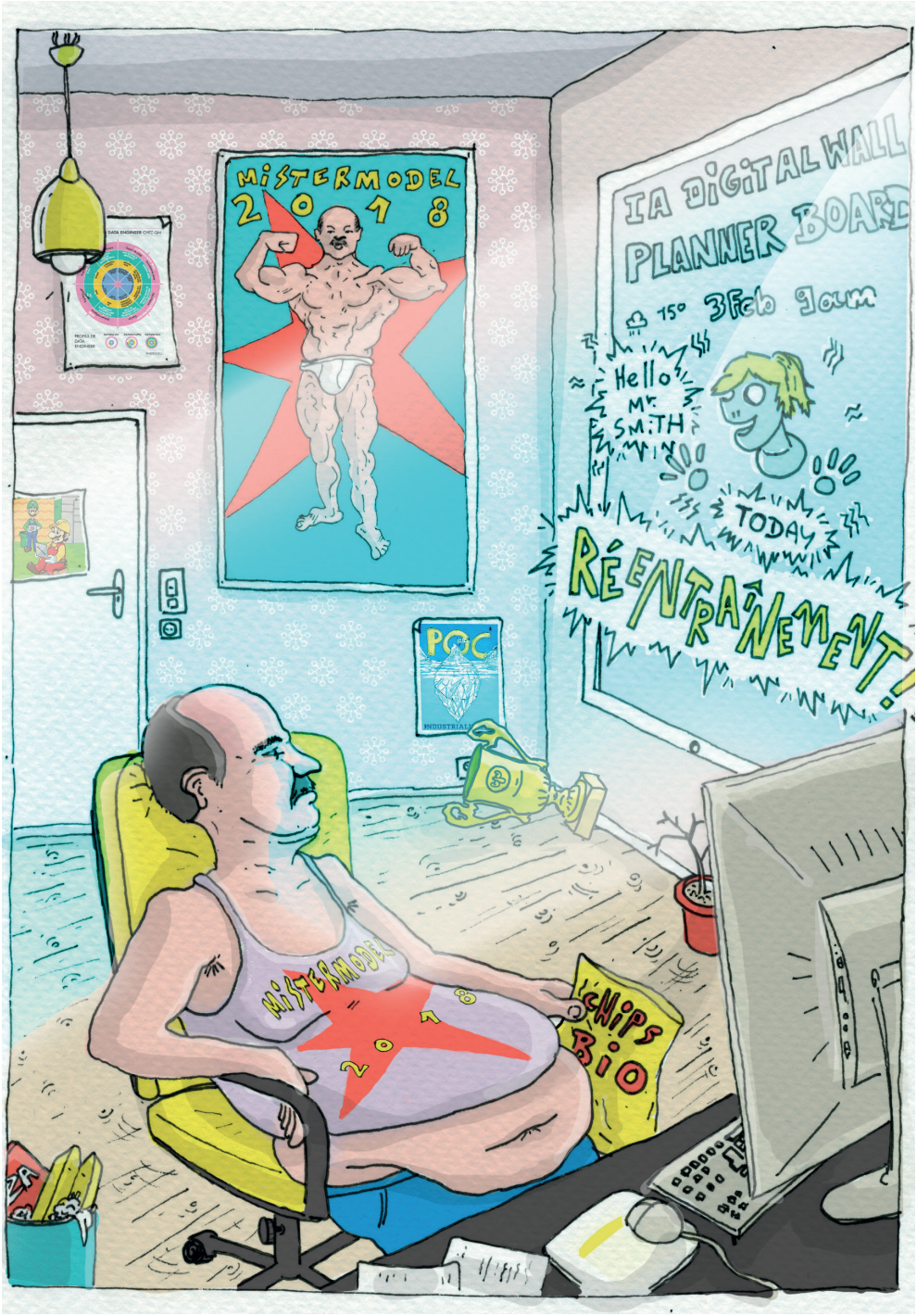
- *Est-ce que le modèle est capable de faire de l'inférence post-entraînement ?* Par exemple, un moteur de recommandation par factorisation de matrice appliqué à un jeu d'interactions historiques [UserID, ItemID] apprend une représentation par userID et par ItemID et ne saura pas faire de prédictions sur des nouveaux clients et produits. De même, un modèle de reconnaissance de logos par classification ne pourra pas identifier des logos non-présents dans le jeu d'entraînement ; si on veut être capable d'identifier de nouveaux logos il faut soit inclure ces nouveaux exemples dans un réentraînement, ou adopter une architecture permettant d'associer des éléments sémantiquement identiques sans nécessairement apprendre à détecter leur individualité. Ce domaine prometteur appelé apprentissage de représentation est en plein essor.

- *Quel est l'impact attendu d'entraîner sur des données plus récentes ?* Par exemple, un modèle de classification de sentiment de texte entraîné sur un corpus de français n'aura peut-être pas de gain de performance significatif si on l'entraîne sur le même volume de données mais plus récent d'un mois ; le français n'a pas forcément beaucoup changé en un mois.

- *Quel est l'impact attendu d'entraîner sur un plus gros volume de données ?* La qualité d'un modèle croit généralement avec la taille du dataset d'entraînement. Néanmoins, est-ce que le bénéfice incrémental d'un réentraînement couvre l'investissement associé ? Passé une certaine taille de dataset il se peut que non.

Enfin, un point important est aussi que certains modèles supportent l'entraînement incrémental ce qui évite de réentraîner sur tout le *dataset* et donc de réduire les coûts et le temps de mise à jour. C'est le cas des modèles de vision intégrés dans SageMaker (détection et classification) mais aussi de certains modèles Scikit-learn qui supportent le « partial_fit ».

Ce qui est intéressant, c'est que lorsqu'on est capable de correctement mesurer le bénéfice d'un réentraînement sur une métrique de son choix, le choix d'une politique de réentraînement pourrait en principe s'apprendre de manière automatisée par optimisation en ligne ou renforcement (e.g. avec en entrée l'âge et la taille du dataset, la déviation des variables depuis la dernière mise à jour), mais je n'ai pas encore vu ce type d'apprentissage en pratique.


MISTERMODEL
2018
IA DIGITAL WALL
PLANNER BOARD
15° 3Feb Jam
Hello
Mr.
SMITH
TODAY
RÉENTRAÎNEMENT!
POC
INDUSTRIAL
MISTERMODEL
2018
CHIPS
BIO

3. EVOLUTIONS PROGRAMMÉES ET GESTION DU RÉ-ENTRAINEMENT

***Contributeurs** : Guillaume Mocquet*

Les diagnostics issus du *monitoring* de modèles, évoqués dans le chapitre précédent, ne peuvent rester sans action. En pratique, il doit naturellement découler de la recherche des causalités d'un problème sur les prédictions du modèle une ou plusieurs modifications portant sur le modèle lui-même, ainsi que ses données d'entrée.

Ce chapitre vise donc à recenser les méthodes et les bonnes pratiques permettant d'accélérer, et pourquoi pas d'actionner le ré-entraînement des modèles dans un projet, et ce à plusieurs niveaux :

- côté code d'abord, en se munissant des outils adaptés, dès le début du projet ;
- côté architecture ensuite, via la mise en place de composants et de processus dédiés ;
- côté gestion de projet enfin, par l'identification et la prise en charge des parties non automatisables.

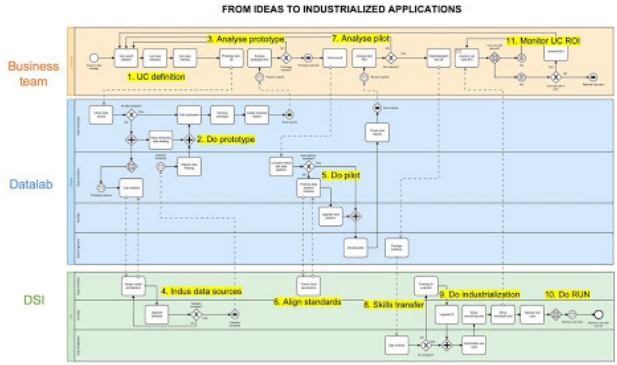
A. ORGANISATION

Avant d'aborder les enjeux techniques, il est important de sensibiliser l'ensemble des membres qui interagissent de près ou de loin avec le(s) modèle(s). L'objectif premier est de **désiloter l'organisation** pour faciliter les tests de nouveaux modèles, augmenter la fréquence des mises en production, réduire les risques lors du déploiement d'une nouvelle version...

Les projets de *Machine Learning* (ML) / *Data Science* (DS) requièrent une multitude de corps de métier qui n'ont pas les mêmes objectifs et responsabilités : Data Architects, Solutions Architects, Data Engineers, Data Scientists, Administrateurs Systèmes & Réseaux, Développeurs Back-end / Front-end / API, équipes métiers, experts DataViz, etc. Dans les grandes organisations, il est fréquent que l'ensemble des parties prenantes au projet de ML/DS ne se parlent pas. De plus, ces dernières sont généralement rattachées à différentes directions, voire réparties sur différents sites géographiques.

La clef de la réussite d'un projet de ML/DS passe par la mise en place d'un **SI agile transverse à l'organisation**. Il est important que l'ensemble des acteurs puissent interagir de façon (relativement) autonome sur l'ensemble des phases de développement, recette, mise en production, etc. La mise en production d'un nouveau modèle ne doit pas dépendre de la disponibilité d'une personne/équipe. Le cas échéant, cela créer un goulot d'étranglement qui ralentit l'actualisation des modèles. Or augmenter la fréquence des mises en production contribue à réduire le niveau de stress / risque des mises à jour.

Pour se faire, il est important de connaître précisément les interactions entre les différentes *business units* avec un focus sur les zones "d'échanges d'actions". La réalisation du **diagramme de collaboration** suivant la norme **BPMN** 2.0 est un excellent point de départ. Ci-dessous, un exemple de modélisation du processus d'industrialisation des cas d'usages de ML/DS au sein d'un grand groupe coté en bourse :



À partir du diagramme de collaboration détaillé, il est facile d'identifier les principales zones fonctionnelles et les interactions entre les différentes *Business Units*. Dans l'exemple ci-dessus, il s'agit des 11 points annotés en noir sur fond jaune. Les interactions entre les business units sont représentées par les flèches en pointillé. À noter la présence d'échanges de messages entre les différentes business units pour se synchroniser aux étapes clés lors de l'avancement des tâches dans le workflow.

Les "piscines" — grandes bandes de couleurs verticales — représentent chaque *Business Unit*. À l'intérieur, les tâches sont décomposées suivant les différents corps de métier : Data Architect, Data Scientist, etc.

Ce diagramme fait ressortir une vérité bien souvent oubliée : le design et le développement des modèles sont réalisés par l'équipe du datalab, par contre le **RUN est opéré par l'équipe de la DSI**. Dans bien des cas, ce sont deux mondes totalement distincts !

Le travail réalisé ci-dessus est un bon point de départ pour définir le workflow de mise en production transverse à l'organisation des cas d'usages à base de ML/DS. Pour aller plus loin dans la gouvernance du cycle de vie des modèles et les actions à mettre en place, consulter le chapitre 6.

B. AUTOMATISATION

La réussite du déploiement des cas d'usages ML/DS réside dans la capacité de l'organisation à **industrialiser et automatiser** ses workflows de mise en production sans créer de goulot d'étranglement. Dans cette optique, il convient d'utiliser les outils utilisés par les principes **DevOps**. Attention toutefois à ne pas tomber dans le dogme ! En effet, contrairement à un projet informatique "standard", les projets de ML/DS requièrent une plus forte implication des utilisateurs métiers. Ainsi, il convient d'être extrêmement **agile** et d'accompagner ces derniers dans la prise de responsabilité du workflow de mise en production. C'est un fort challenge car ces profils sont des experts sur le plan métier et business mais généralement néophytes sur le plan technique. Il est important de décomplexer et automatiser les tâches techniques du workflow. Ce point est paradoxal, car les projets ML/DS requièrent une expertise de plus en plus pointue sur le plan technique.

L'automatisation du workflow est une tâche complexe, qui nécessite de prendre en considération pléthore de sujets :

- Workflow / gestion de projet ;
- L'ossature de l'architecture ;
- Stack CI/CD ;
- Gestion des versions des applications ;
- Gestion des versions des modèles.

WORKFLOW / GESTION DE PROJET

Il est important que l'ensemble des parties prenantes collaborent ensemble via le même outil et le même workflow. Cela passe par l'utilisation d'un outil de ticketing/ gestion de projet sur mesure. Dans les grandes organisations, on retrouve bien souvent le logiciel **JIRA** édité par la société Atlassian. Il s'agit de la référence incontestable en la matière. À l'instar de SAP, c'est un progiciel complexe à appréhender mais nécessaire pour modéliser un workflow digital qui "colle" à la philosophie, aux usages et aux particularités de l'organisation. C'est l'outil qui doit s'adapter à la façon de travailler des collaborateurs et non l'inverse. Utiliser un produit (trop) simple s'avère rapidement contre productif car trop restrictif !

L'automatisation du workflow nécessite d'utiliser une solution de *Continuous Integration* (**CI**) / *Continuous Delivery* et *Continuous Deployment* (**CD**) couplé à l'exploitation des **APIs** JIRA.

L'OSSATURE DE L'ARCHITECTURE

Mettre au point un projet de ML/DS ne doit pas faire perdre de vue les standards mis au point depuis des décennies par l'architecture logicielle "standard". Il n'est pas rare, en début de projet ML/DS, de voir les *Data Scientists* et *Data Engineers* foncer sur le développement du code en omettant les bonnes pratiques du développement logiciel :

- Absence d'**outil de gestion de version** telle la référence **GIT.**
- Non implémentation des **design patterns** favorisant la réutilisation du code sans copier/coller, tels que la **modélisation objet.**
- Application monolithique faiblement **modulaire** : il est difficile de **réutiliser un composant développé dans un autre contexte.** Par exemple, utiliser du code issu d'une application de production en mode exploratoire via un notebook.
- Traitements **automatisés** quasiment inexistants. Dans bien des cas, il s'agit de divers scripts bash et/ou python "bricolés" par divers développeurs / *Data Scientists* sans réelle cohérence ni vision d'industrialisation sur le long terme.

Pour que les applications avec des cas d'usages ML/DS soient en mesure de scaler correctement une fois en production, il est important d'investir dès le début du projet sur les problématiques d'architecture et mettre en place le plus tôt possible les bonnes pratiques évoquées dans le paragraphe précédent. Plus les notions d'architecture sont abordées tardivement, plus la dette technique est importante et dure à supprimer !

Sur les projets informatiques qui manipulent beaucoup de données, il est important de centraliser les *features* / KPIs générés de façon à maximiser la réutilisation du travail réalisé par une équipe. En 2017, Uber théorise le concept de ***Feature Store*** qui permet aux équipes de partager, de découvrir et d'utiliser un ensemble de *features* / KPIs conservés comme point d'entrée d'un nouveau

cas d'usage ML/DS. Ce concept est né du constat que de nombreux problèmes de modélisation chez Uber utilisent des fonctionnalités identiques ou similaires, et qu'il est très utile de permettre aux équipes de partager des fonctionnalités entre leurs propres projets et aux équipes de différentes organisations de partager des fonctionnalités entre elles. Les bénéfices sont doubles :

- Raccourcir les délais de développement des projets ML/DS : capitalisation des traitements.
- Augmenter la qualité et la disponibilité des applications à base de ML/DS : suppression de l'effet copier/coller..

STACK CI/CD

Mettre en place une stack de CI/CD est un point essentiel qui est souvent omis lors de la construction de la roadmap des projets ML/DS car jugé (à tort) "*overkill*". Posséder une telle stack constitue pourtant un **pilier de la philosophie DevOps.** L'outil de prédilection est **Jenkins**. Véritable pionnier, il s'agit d'un produit Open Source fort d'une dizaine d'années d'existence. Cette ancienneté lui confère une communauté riche et active. De plus, ce produit dispose de plus de 1 000 plugins pour s'interfacer facilement avec les principaux outils / technologies du marché : GIT, Docker, unit tests, JIRA, etc.

Suivant le niveau de maturité, le paramétrage de la chaîne de CI/CD peut être réalisée en mode clic bouton ou **intégralement en code** via le langage **Groovy.** Ce dernier point est particulièrement intéressant pour historiser les changements de votre chaîne de CI/CD simplement dans un *repository GIT*, au même titre que l'ensemble des sources du projet.

La bonne pratique repose sur l'utilisation des ***pipelines*** Jenkins. Les grandes étapes du *workflow* sont les suivantes :

1. Checkout de la dernière version de la branche develop
2. Génération du nouveau numéro de version
3. Création d'un nouveau Tag GIT (sans publier sur le remote)
4. Exécution des tests unitaires
5. Exécution des tests d'intégrations

6. Merge la branch develop sur la branche master
7. Commit et publication de la branche master (inclus le Tag GIT précédemment créé) sur le remote
8. Checkout du Tag GIT depuis la branche master
9. Création des artefacts (i.e. build)
10. Publication des artefacts sur le service de repository binaire

À noter : Si les tests unitaires et intégrations ne passent pas, les étapes suivantes ne sont pas exécutées.

Contrairement à l'utilisation des produits proposés par les hébergeurs de solutions cloud tel AWS ou GCP, l'utilisation de **Jenkins évite le *vendor lock-in.*** En effet, les scripts développés sont portables sur n'importe quelle autre stack : aussi bien on-premise que sur un autre hébergeur.

GESTION DES VERSIONS DES APPLICATIONS

Au bout de la chaîne de CI/CD (cf. section précédente), il est important de conserver l'historique des versions générées, dépendances incluses, en vue de **pouvoir rétablir rapidement une ancienne version de l'application**. A première vue, agir de la sorte peut sous-entendre une duplication de l'historique des versions dont le code source est déjà stocké au sein d'un ou plusieurs dépôts GIT. Cependant, tracer les évolutions des versions avec les bonnes versions des dépendances peut vite s'avérer être un enfer !

De plus, un système de gestion des versions binaires prend en charge l'hébergement local des dépendances tierces. Ceci permet de prémunir le SI de la disparition et/ou de l'indisponibilité d'une bibliothèque publiée sur Internet / GitHub.

Pour finir, ce type d'outil permet une gestion accrue des droits. Il est possible de différencier les utilisateurs autorisés à "tirer" les artefacts, des utilisateurs qui publient de nouvelles versions (généralement la machine en charge de la CI/CD). Dans le cas d'un SI composé de plusieurs écosystèmes, il est important de sélectionner un outil qui prenne en charge l'ensemble des technologies utilisées (Python, Docker, Java, ...).

Nexus Repository Manager est une solution de premier choix. Celle-ci est disponible en deux éditions :

- OSS : l'offre de base gratuite (largement suffisant pour démarrer dans de bonnes conditions).
- PRO : un produit payant pour les entreprises en quête de besoin de haut niveau (haute disponibilité, support technique premium, etc.).

Ce produit à l'avantage de gérer la majorité des formats manipulés dans l'écosystème ML/DS tel **Python, Docker, APT**/YUM, Maven/Java, npm, P2 et bien d'autres.
L'administration des utilisateurs est réalisée via une interface graphique simple d'utilisation. Il est possible de configurer un connecteur **LDAP** pour bénéficier du dictionnaire des utilisateurs et groupes déjà en place au sein de l'organisation.
La mise à disposition des divers artefacts applicatifs facilite l'urbanisation et la sécurité du SI :

- Gestion centralisée des accès ;
- Gestion unifiée des différents modules, bibliothèques, et dépendances.

GESTION DES VERSIONS DES MODÈLES

A l'instar de la gestion des versions des sources & artefacts, il est important de procéder de même pour les modèles. L'intérêt principal est identique, à savoir pouvoir (ré-)exécuter un ancien modèle **facilement**. Outre cet aspect, il est intéressant de pouvoir se servir de différentes versions d'un même modèle en **simultané**. Charge à l'application de sélectionner la version du modèle qu'elle souhaite solliciter. C'est un élément essentiel pour automatiser la mise en production des nouveaux modèles publiés au sein du gestionnaire de code source (i.e. **trigger sur un push GIT**) . La **chaîne de CI/CD met automatiquement** à disposition chaque nouvelle version publiée sur la branche **develop** GIT. Chaque modèle est "monitoré", l'application compare les résultats des différents modèles. Dès lors qu'un nouveau modèle possède un meilleur score que le modèle actuellement en production, celui-ci devient alors le nouveau modèle en production. Ce principe est inspiré du mode de déploiement **Canary Release** que l'on rencontre dans l'univers des services web à très fort trafic. Avec l'avènement des containers, ce mode de déploiement est plébiscité par l'écosystème Kubernetes.

Dans l'écosystème ML/DS, ce concept n'est pas récent mais manque de maturité. Il y a encore peu de temps, il n'existait aucun framework Open Source en charge de cette problématique. Ce vide est sur le point d'être comblé par la percée fulgurante de **MLflow**. Il s'agit d'une solution Open Source éditée par **Databricks,** acteur de référence dans l'écosystème Big Data. Ce dernier est principalement connu pour éditer des solutions autour de la base de données **Cassandra**. En avril 2019, Microsoft intègre le projet MLflow et ajoute un support natif à ce produit au sein d'Azure ML. La version 1.0.0 est toute récente, en date du 13 juin 2019 !

Avec MLflow, les *Data Scientists* peuvent suivre et partager des expériences localement (sur un ordinateur portable) ou à distance (dans le *cloud*), regrouper et partager des modèles à travers des frameworks et déployer des modèles pratiquement partout.

MLflow est organisé en trois composantes : **Tracking, Projects et Models.** Chaque composant est utilisable séparément - par exemple, l'export des modèles au format de modèle de MLflow ne nécessite pas Tracking ou Projets - mais ils sont également conçus pour bien fonctionner ensemble.

La philosophie de base de MLflow est d'imposer le moins de contraintes possibles au *workflow* : il est conçu pour fonctionner avec n'importe quelle bibliothèque d'apprentissage machine, déterminer la plupart des choses sur le code par convention, et nécessiter un minimum de modifications pour s'intégrer dans une base de code existante. En même temps, MLflow vise à rendre reproductible et **réutilisable** par de multiples *Data Scientists* n'importe quelle base de code écrite dans son format.

QUESTIONS À ARDUINO CASCELLA (DATABRICKS)

Arduino Cascella est Solutions Architect à Databricks et engagé dans la communauté autour de Spark, MLflow et Delta Lake. Avant de rejoindre Databricks, Arduino a travaillé en tant que Data Scientist, en particulier dans la mise en production de solutions de Machine Learning pour la détection de fraude.

Quel typologie de métier data est concerné par MLflow ? Tous les acteurs ? Ou bien un métier particulier (MLEngineer) ?

L'objectif de MLflow est de fluidifier tout le cycle de développement d'outils ML, centraliser et partager l'information sur les entraînements notamment entre tous les acteurs du cycle de vie des modèles.

Les profils concernés sont surtout les équipes Data Science, conceptrices du modèle, et MLEngineer pour la mise en production et la maintenance. MLflow permet à la fois de packager facilement le code (notamment la gestion des versions pour faire des prédictions et déployer), et d'assurer la reproductibilité du modèle ainsi qu'une mise en production transparente pour les utilisateurs. Il est cependant moins à destination des Data Analysts.

Deux problématiques de plus en plus récurrentes chez nos clients sont la gestion du ré-entraînement des modèles à certaines échéances, ainsi que le monitoring du modèle déjà déployé. MLflow adresse-t-il ces sujet ? Si oui, comment ?

MLflow joue un rôle principalement jusqu'au déploiement du modèle, mais il est courant que les utilisateurs loggent dans MLflow des métriques de déploiement. La mise en place de calcul de KPIs de monitoring ML à proprement parler doit être prise en charge par l'utilisateur.

Concernant le réentraînement, MLflow résout la reproductibilité, car le modèle est packagé avec les données, en artefact ou via une référence à une version spécifique des données dans Delta Lake. Il permet également de tracer les entrainements/ré-entrainements, tout en pouvant être intégré dans des jobs Databricks, ou encore avec un opérateur Airflow.

Dans les versions futures, il est envisagé d'intégrer une telle brique, le "Model Registry" qui offrira un workflow CI/CD permettant le

déploiement automatique de modèles en fonction de tests de qualité sur la sortie des modèles. Ce sujet est bien sûr complexe en raison de la variété des besoins, selon les cas d'usage et les utilisateurs.

Delta Lake, lancé en open source par Databricks au Spark Summit d'avril 2019, est une surcouche de stockage qui fournit des transactions ACID pour les Data Lake, via les APIs d'Apache Spark. L'objectif de Delta Lake est de devenir le format de stockage standard pour le Big Data, amenant des fonctionnalités natives de versioning de données et des garanties sur la qualité des données. On a ainsi un log des commits pour garantir les transactions ACID, la possibilité de faire des rollback, le tout étant appuyé sur des fichiers parquet. Nous pensons que c'est une évolution cruciale pour permettre la fiabilisation du pipeline de données des modèles en production.

Si l'on veut déployer un pipeline en production, avez vous des recommandations ? Par exemple, séparer les étapes de prétraitement du modèle en lui-même en les exposant sur deux points d'entrée différents ?

Tout à fait. Il faut décomposer le pipeline (données et ML) en tâches unitaires, les plus petites possibles, afin de bien tester chaque brique indépendamment des autres. Il est à noter que nous préconisons d'encapsuler la logique de pre-processing de données dans le modèle, avec par exemple MLeap qui permet d'inclure des pre-processing Spark dans le modèle, qui est la dernière étape du pipeline.

Aujourd'hui il faut déployer les différentes étapes du pipeline de façon indépendante, mais nous sommes en train de réfléchir à la meilleure façon de représenter les dépendances et fixer des contraintes de qualité de données de façon plus formelle afin de les intégrer avec Delta Lake.

Il est à noter qu'une des prochaines fonctionnalités de Delta Lake permettra de définir des "expectations" afin de contrôler de façon très fine la qualité des données écrites, refusant au besoin des entrées non conformes. Ceci permet de s'assurer de la qualité des features d'un modèle pour n'exécuter le pipeline que si les conditions sont remplies.

QUESTIONS À PIERRE ARBELET (ALKEMICS)

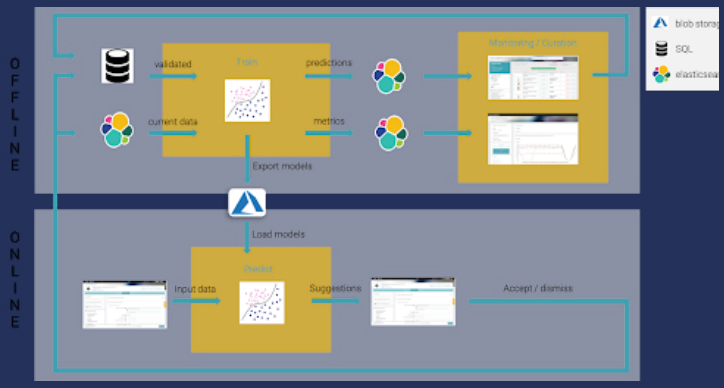
Pierre Arbelet est Lead Data Scientist chez Alkemics depuis 4 ans. Alkemics est une plate-forme B2B où les distributeurs, les marques et les applications partagent et travaillent sur la donnée produit. Avoir une donnée produit précise, complète et à jour est essentiel tout au long du cycle de vie d'un produit.

Pour aider les distributeurs et les marques à obtenir la meilleure qualité de données en terme d'exactitude et de complétion, tout en facilitant la création de contenu, ils ont créé un framework de suggestions automatiques permettant de suggérer en temps réel des informations sur le produit dans la plate-forme, en indiquant les erreurs potentielles détectées dans les données du produit ou en aidant l'utilisateur à saisir les bonnes informations.

Quel pipeline technique de réentrainement avez vous mise en place?

Le workflow de suggestion est composé de deux parties:

- Offline: entraînement et mise à jour des modèles, monitoring et prédictions batch
- Online: serving des modèles et prédictions temps-réel, monitoring des interactions utilisateurs



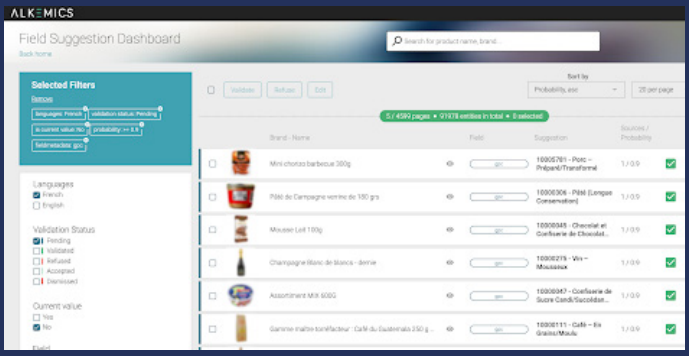
Le workflow offline est orchestré par Luigi (https: /github.com/spotify/luig), un outil qui permet de créer des pipelines de jobs batch, de même qu'un AirFlow. Il est conçu pour être facilement configurable et agnostique du framework ou de l'algorithme Python.

Les modèles sont ré-entraînés chaque jour pour tenir compte des derniers ajouts et modifications de la donnée, récupérée d'un index ElasticSearch et de bases de données SQL.

La donnée d'entraînement est constituée de la donnée de produit, telle que présente en production, associée à des métadonnées issues d'un dashboard interne ainsi que des interactions utilisateurs avec les suggestions.

- Ce dashboard centralise l'ensemble des suggestions des modèles et permet de:
- *Visualiser* les prédictions de classes ayant de mauvaises performances et comprendre les causes des erreurs de classification.
- *Corriger* les erreurs de classification via le dashboard en assignant les étiquettes appropriées. Se concentrer sur les prédictions ayant de faibles probabilités permet d'avoir le plus fort impact sur la performance des modèles, car elles se situent généralement près des limites de décision.
- *Initialiser* des classes à l'aide de la barre de recherche en assignant des étiquettes aux éléments sélectionnés. Les ensembles d'entraînement peuvent ainsi être créés ou améliorés très rapidement.
- *Comparer* des prévisions et des données actuelles pour identifier rapidement les données manquantes ou erronées dans un assortiment d'utilisateurs.

Toutes les étiquettes créées, validées ou modifiées de façon interne ainsi que celles refusées par les utilisateurs sur la plateforme sont ensuite prises en compte lors de la construction de l'ensemble d'entraînement lors des prochaines itérations du workflow offline. Cette boucle de rétro-action s'avère particulièrement efficace.



Les modèles entraînés sont stockés et historisés sur Azure Blob Storage, un service de stockage cloud de donnée non structurée.

Un service python est responsable du serving des derniers modèles entraînés, mis en cache mémoire après téléchargement des serveurs de stockage de sorte qu'ils soient directement disponibles sur le serveur du service. Les prévisions sont effectuées à l'aide d'API dédiées: sur la base de données textuelles en entrée, le service instancie les modèles, fait des prédictions et les structure pour le front-end.

2) Quel suivi du ROI de vos modèles?

Le ROI est mesuré tout d'abord en terme de *qualité de donnée* et *de taux de complétion:* évolution du nombre de produits avec une donnée satisfaisant aux règles de validation. Ensuite en terme de t*aux d'acceptation* et de *gain de temps* utilisateur: à chaque acceptation ou refus d'une suggestion, un évènement de tracking est généré, l'ensemble étant monitoré avec MixPanel.

3) Quelles sont vos bonnes pratiques de la gestion du cycle de vie des modèles?

- Avant d'intégrer un modèle en production, s'assurer de la qualité de l'ensemble de l'infrastructure, de la collecte de la donnée à l'affichage des prédictions et à la boucle de rétroaction. Il est même possible, voire préférable, de commencer sans modèle, avec de simples heuristiques, pour tester l'infrastructure de bout en bout : s'assurer de la bonne couverture des tests (unitaires et d'intégration), du monitoring de toutes les métriques et du bon fonctionnement de l'ensemble de la chaîne de traitement.

- Détecter les problèmes et/ou dégradation des performances des modèles mis à jour avant de les exporter en production: décorréler le workflow batch du serving des modèles.

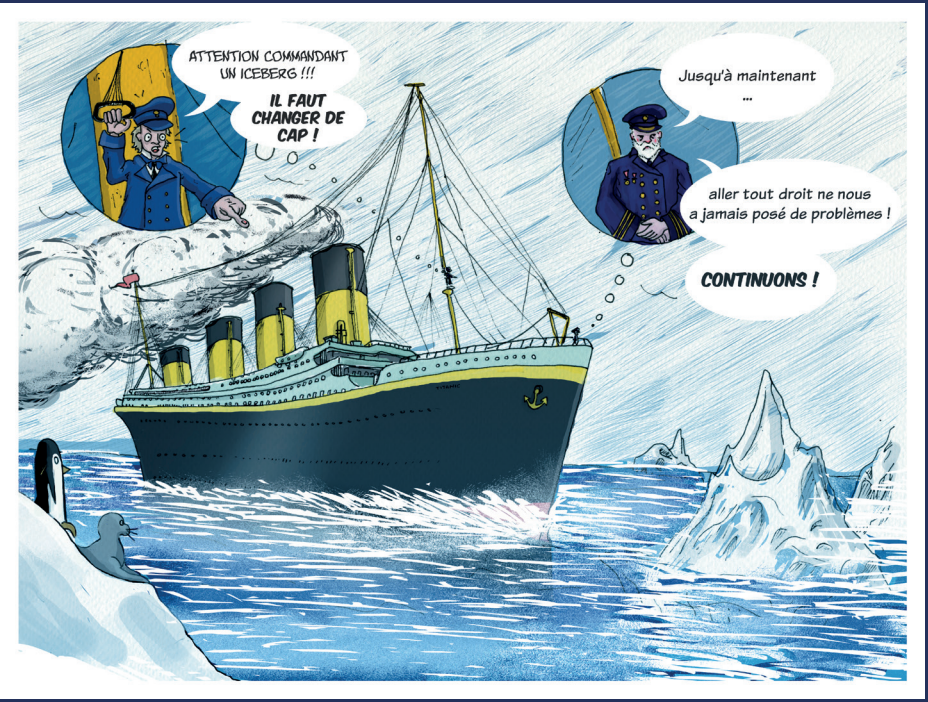
- Historiser les versions des modèles pour pouvoir rollback le cas échéant.

- Mettre en place un monitoring et un alerting efficaces, ne pas suivre uniquement les performances des modèles mais aussi l'évolution des interactions utilisateurs avec les prédictions.

- Définir la fréquence d'actualisation des modèles en fonction du problème et de la vitesse d'évolution des données d'entraînement: compromis coût / dégradation des performances.

4. CYCLE DE VIE ET DÉRIVE DES MODÈLES

Contributeurs : *Antoine Charlet, Gaultier Le Meur, Grégoire Martinon*



A. LE SYNDROME DE LA PERSISTANCE RÉTINIENNE INFINIE

Si vous fixez les 3 petits points blancs situés au milieu de l'image pendant environ 25 secondes puis regardez une surface blanche, vous y verrez très certainement un visage s'y dessiner. Ce laps de temps pendant lequel l'image semble s'être figée sur notre rétine illustre une propriété de l'oeil bien connue utilisée dans le cinéma que l'on appelle persistance rétinienne : c'est ce qui rend l'image fluide. En quoi cela concerne notre problématique de dérive ? Et bien voilà, un modèle que l'on entraîne statiquement sur un flux de données à un instant t, c'est un peu comme un oeil qui souffrirait d'une persistance rétinienne infinie. Si vous vous demandez ce que vous y verriez après avoir regardé un film comme Rocky avec un tel oeil, voici ce à quoi cela pourrait ressembler (1) :

Plutôt artistique n'est ce pas ? Cela dit, il faut bien admettre que si votre tâche avait été d'identifier le nom du film d'origine uniquement à partir de cette image, cela n'aurait certainement pas été une mince affaire, alors que le visionnage du

film en entier vous aurait certainement permis de le faire sans trop de difficulté.

L'idée à retenir est que rien ne dure éternellement. Ainsi, même les modèles les plus sophistiqués construits sur des jeux de données de qualité verront leur performance prédictive décroître au cours du temps. Les données évoluent au fur et à mesure, de nouvelles catégories apparaissent, les distributions changent, même notre interprétation des données change.

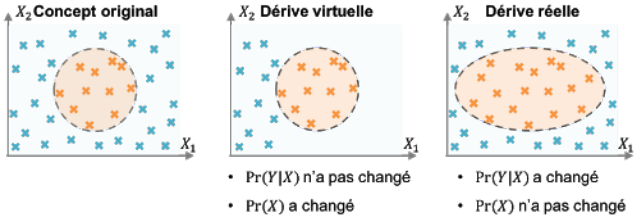
B. LE MONDE ÉVOLUE, POURQUOI PAS LES DONNÉES ?

La grande majorité de la littérature sur l'apprentissage machine part d'une hypothèse commune : **la stationnarité de la distribution des données**. Or, les envies des consommateurs changent avec les saisons, les fraudeurs inventent de nouvelles techniques pour outrepasser les barrières anti-fraude, les spammeurs trouvent de nouvelles tactiques pour atteindre nos boîtes mail… Comment faire pour mettre en production des modèles assurant une performance constante des systèmes de ciblage marketing, anti-fraude ou anti-spam ?

Un concept est la relation entre les variables d'entrée (*features*) et la variable cible (*target*), c'est la distribution de probabilité jointe. C'est ce que l'apprentissage machine s'efforce de modéliser (2). Le théorème de Bayes peut être utilisé afin de décomposer un concept en une probabilité conditionnelle (aussi appelée vraisemblance statistique) et une probabilité *a priori* :

$$C_t = Pr(X,Y) = Pr(Y|X)\ P(X)$$

On parle alors de dérive dès lors que $C_t \neq C_{t+1}$. Quand un concept varie, la variation est due à la probabilité a priori – appelée dérive virtuelle – ou à la probabilité conditionnelle – appelée dérive réelle. Dans le cas où les deux varient en même temps, on parle toujours de dérive réelle.



LES VARIABLES ÉVOLUENT

Intéressons-nous au type de dérive le plus courant : la dérive virtuelle (*covariate shift*). On peut retrouver ce type de dérive dans la différence des distributions entre le jeu d'entraînement et le jeu de test, mais surtout dans des situations de la vie courante. Par exemple, on s'attend à ce que le comportement des utilisateurs d'une carte bancaire change de manière imprévisible : soldes, célébrations, changements de situation, vacances…

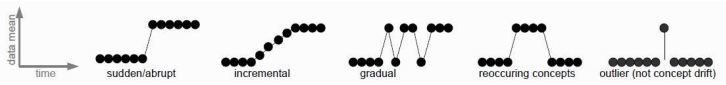
LA CIBLE D'APPRENTISSAGE ÉVOLUE

Le second type de dérive concerne la variation temporelle de la distribution conditionnelle *Pr(Y\X)* ; elle est appelée dérive réelle (*concept shift* ou *dataset shift*). Une illustration de cette dérive est l'évolution temporelle de la résistance aux antibiotiques des souches pathogènes (3).

Dans la grande majorité des cas, c'est bien le manque d'information qui fait que nous ne pouvons pas anticiper un changement qui serait évident si nous avions le bon niveau de connaissance. On appelle ce manque d'information le contexte caché (*hidden context*) (4). Par exemple, dans un contexte d'attrition client B2B (business-to-business), le contexte caché des entreprises étudiées évolue par le biais des fusions acquisitions, et l'indépendance des observations est soumis aux fluctuations du marché. De même, les évolutions des standards IT ne sont pas forcément communiqués aux concepteurs de modèle, qui ne peuvent que constater une dérive de données (changement d'unité dans une mesure par exemple). Enfin, un modèle pilote est en général soumis à une dérive quand on déplace ou étend son périmètre, car on le confronte à des données qui sont *a priori* distribuées différemment.

L'IMPORTANCE DE LA VITESSE D'ÉVOLUTION

Afin de s'intéresser à la détection d'une dérive, il est primordial de prendre en compte la temporalité et la répétition des variations des distributions. On peut imaginer une variable continue générée par une source, par exemple une variable gaussienne de moyenne μ. Les différents types de dérive correspondent à différentes évolutions possibles de la source (2) :



› **Dérive abrupte** : μ passe subitement d'une valeur $\mu 1$ à une valeur $\mu 2$ distincte. Par exemple, une période de soldes va perturber soudainement les habitudes de consommation ou les appétences produits.

› **Dérive incrémentale** : μ se déplace lentement dans l'espace des variables explicatives. Par exemple, une caméra de voiture autonome se salit au cours du temps. Les images passent progressivement d'un état "propre" à un état "sale". Idem pour la pluie, le jour et la nuit etc.

› **Dérive graduelle** : μ passe de manière aléatoire d'une valeur $\mu 1$ à une valeur $\mu 2$ distincte avant de se stabiliser. Par exemple, une entreprise française décide de monter à l'international et de rédiger tous ces mails en anglais. Pendant une phase transitoire, il y aura des mails en français et en anglais, puis à terme que des mails en anglais.

› **Dérive récurrente** : μ prend des valeurs $\mu 1$ déjà rencontrées dans le passé, mais pas forcément de manière périodique. Par exemple, les fraudeurs peuvent recourir, à n'importe quel moment, à d'anciennes méthodes de fraude, par vagues, en pariant sur la volatilité de la vigilance des experts sécurité.

À noter qu'il faut distinguer une dérive de concept d'un point aberrant (*outlier*) dont l'apparition est exceptionnelle et ne marque pas un changement durable de distribution.

C. DÉTECTER LA DÉRIVE

Le monde dans lequel nous vivons est en constant changement. Il semble donc inéluctable qu'un modèle statique créé à partir de données historisées devienne rapidement un modèle peu fiable. Pour autant, derrière ce qui pourrait sembler être une évidence se cache un problème non-trivial : détecter une dérive. Néanmoins, la maturité grandissante des entreprises en matière de stratégie data-driven devrait désormais déclencher une prise de conscience face à ces problèmes de dérive et ainsi s'emparer d'un sujet qui est longtemps resté comme confiné à l'intérieur du monde académique.

DIFFÉRENTES MANIÈRES D'APPRENDRE

BATCH LEARNING

L'apprentissage en configuration batch est une approche dans laquelle les données sont dans un premier temps collectées puis éventuellement labellisées. La construction du modèle s'effectue dans un second temps grâce aux données historisées. C'est l'approche la plus simple à mettre en place. Néanmoins, cela implique la création d'un modèle statique qui, inéluctablement, verra ses performances se dégrader plus ou moins rapidement sur des distributions non stationnaires.

Il semble donc nécessaire de construire des modèles capables de s'adapter à ces changements se produisant au cours du temps. Une première solution pourrait être de ré-entraîner régulièrement le modèle sur l'ensemble des données pour pallier les chutes de performance. Néanmoins, cette démarche présente des inconvénients. L'apprentissage peut s'avérer être long et coûteux en ressources informatiques et cela peut parfois être impossible dans la pratique si l'on doit gérer un volume de données trop important pour être stocké en mémoire. La solution de facilité est alors de ré-apprendre notre modèle uniquement sur les données les plus récentes, c'est l'approche fenêtre glissante. Par contre, cette méthode présente un inconvénient majeur : le modèle peut souffrir d'amnésie en oubliant ce qu'il avait appris précédemment.

INCREMENTAL LEARNING ET ONLINE LEARNING

L'apprentissage est dit incrémental lorsque le modèle se ré-entraîne périodiquement sur des nouveaux batchs de données labellisées. Le jeu d'entraînement peut soit croître dans le temps (fenêtre incrémentale), soit être de taille fixe, auquel cas on s'entraîne sur les données les plus récentes (fenêtre glissante).

Dans le cas de flux denses et rapides de données, on peut considérer l'apprentissage en-ligne : l'entraînement permet de mettre à jour les paramètres du modèle à chaque nouvelle observation labellisée à notre disposition.

ADAPTIVE LEARNING

L'entraînement est dit adaptatif lorsque le modèle inclut un module permettant de détecter une dérive et de réagir de manière autonome. Nous présenterons

plus tard différents types d'algorithmes permettant d'effectuer cette détection. La détection de dérive permet à l'apprenant de s'adapter aux données qui ne cessent d'évoluer. A noter qu'un algorithme d'apprentissage adaptatif peut suivre les principes de l'apprentissage en-ligne ou de l'apprentissage incrémental, en fonction du scénario et de la méthode de détection de dérive choisis.

DÉTECTION SUPERVISÉE OU NON-SUPERVISÉE

Un des prérequis en apprentissage automatique est de s'assurer que les données ayant servi à l'entraînement du modèle proviennent de la même distribution que les données sur lesquelles le modèle va inférer. On peut alors enquêter sur la base de données uniquement pour vérifier cette hypothèse, c'est la détection non-supervisée.

Un apprentissage sur un flux de données non-stationnaire aura donc pour conséquence la dégradation des performances du modèle dans le temps, à moins que la dérive ne soit modélisable (prévision de séries temporelles). Une idée fondamentale dans le domaine de la détection de dérive est ainsi de surveiller la qualité de prédiction du modèle en place. On parle alors de détection supervisée. Cette étape cruciale doit être traitée avec rigueur : typiquement les métriques de performances doivent être déclinées sur des sous-populations pour éviter les biais rendus invisibles par effets de moyenne. De plus, la notion de performance doit être clairement définie, et en lien étroit avec le ROI du modèle. Enfin, la qualité d'une prédiction ne se mesure pas que dans sa valeur mais également dans son incertitude. Si l'incertitude du modèle grandit anormalement, c'est aussi une dérive à prendre en compte.

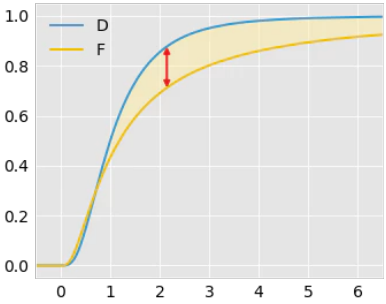
Cependant, Il faut garder à l'esprit que si l'on se place dans une configuration où l'apprentissage s'effectue au fur et à mesure du temps, l'hypothèse selon laquelle les labels seraient disponibles dès l'arrivée d'une nouvelle instance est loin de correspondre à la majorité des cas d'usages rencontrés (voir chapitre 2, délai d'obtention de la *ground truth*). La surveillance du modèle passe donc non seulement par la surveillance des performances mais également par la surveillance des distributions de données, afin de détecter au plus vite les problèmes, sans attendre la labellisation.

COMMENT SAVOIR SI MON JEU DE PRÉDICTION RESSEMBLE À MON JEU D'ENTRAÎNEMENT ?

Quand on se pose la question de savoir si deux échantillons de données ont été générés via le même processus, l'utilisation d'un test statistique est souvent la première idée qui vient à l'esprit. On peut souvent reformuler le problème en terme de distance entre ces échantillons, ou plus généralement de tests d'appariements. Il en existe une grande variété et nous en présentons quelques-uns ci-après.

LES DISTANCES DE KOLMOGOROV-SMIRNOV ET CRAMER VON MISES

La distance de Kolmogorov-Smirnov (KS) est inspirée du test statistique éponyme. Le calcul de cette distance demande de construire *pour chacune des variables explicatives* (de manière univariée) la fonction de répartition empirique des deux échantillons à comparer. La distance KS est définie comme étant la plus grande différence absolue entre ces deux courbes pour l'ensemble des points. Une propriété utile tant que l'on s'intéresse à la similarité entre deux jeux de données est que cette distance est toujours comprise entre 0 et 1. La distance de Cramer von Mises est obtenue de manière similaire en effectuant cette fois la somme au carré des différences entre les deux fonctions de répartition empiriques. Ce mode de calcul rend la distance de Cramer plus robuste aux déformations locales en comparaison à la distance KS.



Sur cette figure, on représente la fonction de répartition de deux distributions différentes. La distance de Kolmogorov-Smirnov est indiquée en rouge, c'est l'écart maximum entre les deux courbes. La distance de Cramer Von Mises s'obtient elle en sommant les écarts quadratiques entre les deux courbes.

LES DISTANCES D'ÉNERGIE

Une des limitations des distances exposées ci-dessus est qu'elles requièrent d'être calculées pour chacune des variables indépendamment des autres. Une autre approche consiste à utiliser la distances d'énergie (5) sur l'ensemble des variables

(test multivarié). Elle présente l'avantage d'être à la foi invariante d'échelle et par rotation, et permet d'attaquer des distributions multi-dimensionnelles. Soient X et Y deux matrices aléatoires (jeux de données), la distance d'énergie est définie comme étant $2E(\|X-Y\|)-E(\|X-X'\|)-E(\|Y-Y'\|)$ avec X', Y' respectivement une copie de X et Y. Cela permet de calculer directement la distance entre deux jeux de données, et est implémentée dans le package dcor (6).

UN EXEMPLE D'APPROCHE ML POUR UNE SÉLECTION DE FEATURES ROBUSTE À LA DÉRIVE

Plaçons-nous dans un scénario dans lequel nous avons à notre disposition deux batchs de données correspondant respectivement à des instants *t* et *t+1*. Nous souhaitons évaluer si la distribution de nos variables prédictives a évolué entre ces deux instants et ainsi détecter la dérive si elle existe. L'idée est que nous pouvons pour cela labelliser nos données en fonction de leur appartenance à l'un ou l'autre de ces deux batchs et entraîner un modèle initial à discriminer nos instances selon le batch dont elles proviennent. En pratique, l'implémentation peut s'effectuer de deux façons différentes.

Le package MLBox (7) prend le parti de construire un modèle univarié pour chaque variable dont on dispose pour en mesurer la propension à dériver (typiquement via une AUC). On peut ensuite entraîner à nouveau notre modèle initial en écartant les variables dérivant le plus. Cependant, une solution plus directe consiste à ajouter une couche d'interprétabilité à notre modèle initial (voir livre blanc "IA explique-toi !"). Ainsi on peut mesurer directement l'impact de chaque variable dans la capacité du modèle à inférer l'origine de nos instances via des méthodes telles que la *permutation importance*. Cette seconde méthode permet de conserver les effets d'interactions entre prédicteurs mais se fait au détriment d'un temps de calcul accru.

Dans les deux cas, nous cherchons à supprimer les prédicteurs pour lesquels la distribution sous-jacente a évolué de manière significative afin d'améliorer la robustesse de notre modèle. On cherche donc un sous-espace orthogonal à la dérive.

LE CAS DES FLUX DE DONNÉES

MÉTHODE D'ANALYSE SÉQUENTIELLE

Le *Sequential Probability Ratio Test* est un test d'hypothèse séquentiel servant de brique de base pour de nombreux algorithmes de détection de dérive. Supposons que nous ayons à notre disposition une séquence d'observations indépendantes $X= (X_1, ..., X_w, ... X_n)$ pour laquelle $(X_1, ..., X_w)$ et $(X_{w+1}, ..., X_n)$ proviennent respectivement d'une distribution P_0 et P_1. A partir du point w, la probabilité d'observer certaines sous-séquences sous P1est donc significativement plus élevée que sous P0. Par significativement, il faut comprendre que la somme cumulative du rapport des deux vraisemblances ne descend pas en dessous d'un certain seuil choisi. Dans son sillage, on peut également mentionner le test de Page-Hinkley, qui reprend l'idée d'un seuil de dépassement appliqué à une somme cumulée d'écarts à la moyenne.

MAÎTRISE STATISTIQUE DES PROCÉDÉS

La maîtrise statistique des procédés a initialement été utilisée comme outil de contrôle de la qualité des processus industriels. La méthode consiste à comparer la performance du modèle entre une fenêtre de temps de référence et la fenêtre de temps courante. Si à un instant t, on considère que l'erreur a augmenté de façon significative par rapport aux exemples passés, on peut affirmer avec une confiance plus ou moins élevée que la distribution des variables a changé.

SURVEILLER LES DISTRIBUTIONS SUR DEUX FENÊTRES TEMPORELLES

Un algorithme emblématique de ce type d'approche est ADWIN (8). L'idée est de tester si la distribution dont sont issues nos variables prédictives est restée inchangée au cours du temps. Pour cela, un certain nombre d'observations passées indexées par le temps sont conservées en mémoire dans une structure de donnée de type "First In First Out" que nous appellerons la fenêtre courante. On suspecte qu'une dérive a pu avoir lieu à n'importe quel instant t' de notre fenêtre courante. On construit alors une sous-fenêtre gauche contenant toutes les observations antérieures à t', et une sous-fenêtre droite contenant toutes les observations postérieures à t'. Un test d'homogénéité de Hoeffding est effectué pour tous les t' jusqu'à ce qu'un écart significatif entre la moyenne de gauche et de droite soit trouvé.

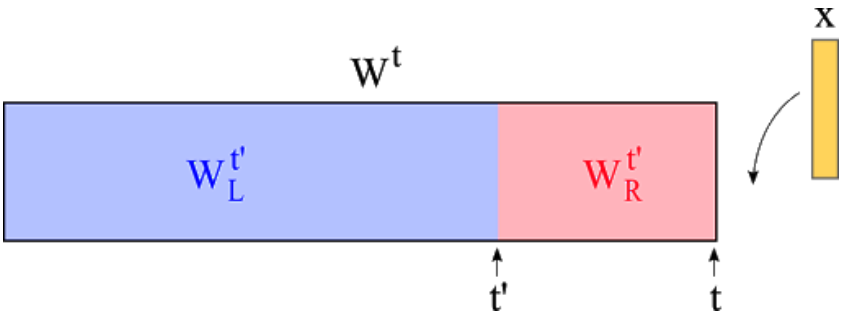


Illustration du détecteur ADWIN. L'historique est séparé en 2 sous-fenêtre gauches et droite, et un test d'inhomogénéité est effectué entre les deux. Dès que le test est positif, on supprime la partie antérieure (gauche). L'intérêt de l'algorithme ADWIN réside dans sa capacité à tester tous les cas possibles en temps logarithmique.

MÉTHODES SE BASANT SUR DES HEURISTIQUES

Le but de ces approches dites contextuelles est d'identifier des fenêtres de temps où la distribution des variables est stationnaire. Pour cela on peut choisir d'enrichir l'espace de représentation des données en horodatant nos observations. Si cette nouvelle variable explicative permet à un arbre de décision de partitionner l'espace selon l'horodatage, cela induit la présence de contextes distincts dans le temps et donc d'une dérive.

D. ACTION/RÉACTION : S'ADAPTER À LA DÉRIVE

« Une solution unique n'est guère possible et n'est pas souhaitable pour la gestion de la dérive conceptuelle » (9). En effet, il n'existe pas de solution optimale et suffisamment générique pour traiter l'ensemble des problématiques de l'adaptation. C'est pourquoi il est nécessaire de diversifier les approches pour identifier la solution pertinente pour chaque problème particulier.

LES MÉTHODES D'ADAPTATION

La publication très complète (10), complétée par (2), présente les différentes approches afin de s'adapter à la dynamique inconnue des données, Il énumère quatre briques fondamentales à considérer : la mémoire, la détection de dérive (*cf.* point 3 ci-avant), l'apprentissage et l'assemblage de modèles.

Ainsi, au-delà de la dérive des données, il existe un objectif de détection et d'adaptation à la dérive du signal, et donc à la métrique de performance du modèle en production. Les méthodes d'adaptation discutées ci-après peuvent être réparties en différentes catégories :

- la gestion de la « mémoire » : *Online vs Batch* ;
- le type de détection : Active (*informed*) vs Passive (*blind*) ;
- le nombre de modèles : Unique vs Ensemble ;
- la gestion du modèle : ré-entraînement vs incrémental.

LA MÉMOIRE ET LE COMPROMIS STABILITÉ/PLASTICITÉ

Le premier bloc qui nous intéresse est donc la gestion de la « mémoire » du modèle. Cela représente la quantité d'information nécessaire pour le (ré) entraînement d'un modèle.

D'une manière générale, un modèle ne doit pas apprendre sur un espace temporel trop grand au risque d'apprendre des concepts qui ne sont plus à jour. Des mécanismes d'oubli peuvent alors être mis en place :

- l'oubli abrupt, où, à chaque itération, la fenêtre glissante ou l'échantillonnage évolue i.e. chaque élément est inclus ou non dans la fenêtre d'entraînement ;
- l'oubli graduel, où, à l'image d'une décroissance exponentielle/linéaire, chaque exemple en mémoire se voit associé à un poids qui décroît en fonction de son ancienneté (ce type d'oubli est applicable sur une fenêtre glissante).

La taille de la fenêtre d'apprentissage influe fortement sur la capacité du modèle à apprendre de nouveaux concepts. Le problème sous-jacent est le **dilemme de Stabilité-Plasticité** (2). En effet, si la fenêtre est trop petite, l'apprentissage du modèle sera aliéné par la grande proportion de bruit dans les données, le modèle a une plasticité trop importante. Si elle est trop grande, il lui sera impossible d'apprendre de nouveaux concepts, empêché par l'inertie des anciens concepts appris précédemment : il est trop stable.

LA PHASE D'APPRENTISSAGE PUIS D'ADAPTATION

Le second bloc est l'apprentissage du modèle, ou plutôt la réinitialisation de sa mémoire après une dérive. En première approche, cette phase peut être réalisée en supprimant et en ré-entraînant intégralement le modèle. D'autres modèles non-standards peuvent évoluer selon leurs propres mécanismes d'adaptation. C'est le cas de certains arbres de décision : quand la dérive n'a concerné qu'une partie finie de l'espace des données, le remplacement peut être local : uniquement les nœuds impactés par la dérive seront modifiés. Un exemple connu d'arbre adaptatif est l'Arbre de Hoeffding adaptatif (11) dont une implémentation est disponible dans la librairie scikit-multiflow (12). Il est notamment pensé pour se mettre à jour de manière incrémentale et s'adapte bien aux flux de données variables dans le temps.

D'une manière générale, l'adaptation d'un modèle prédictif peut être réalisée de deux manières :

- passive (*blind*) : le modèle est adapté de manière périodique, sans connaissance d'une chute de performance quelconque.
- active (*informed*) : la stratégie d'adaptation du modèle dépend d'un événement déclencheur (principalement d'une dérive dans les données ou d'une chute de performance).

LES COMITÉS D'EXPERTS

Les ensembles de modèles, aussi appelés comités d'experts, semblent être la solution à considérer dans de nombreuses situations (13). En effet, à l'instar des forêts aléatoires, pour apprendre dans un environnement non-stationnaire, un ensemble pourra compter sur l'intelligence collective afin de prédire et s'adapter dynamiquement par ajout, suppression ou modification des modèles. L'idée générale est d'avoir une panoplie de modèles entraînés sur des périodes de temps différentes, un peu comme des historiens sont experts de différentes parties de l'histoire. Lorsqu'une nouvelle donnée arrive, c'est l'expert le plus pertinent pour le concept en cours qui est actionné.
Pour prédire, les comités d'experts se basent donc sur des techniques de « vote » et d'adaptation différentes :

- une moyenne pondérée des prédictions, la pondération dynamique étant basée sur les performances de chaque modèle : plus le modèle est performant, plus sa voie sera importante ;

› une majorité sans pondération avec la mise en place d'entraînements de nouveaux modèles qui remplacent les moins performants ;

› la prédiction du modèle le plus performant

LA DUALITÉ ENTRE TEMPS D'ADAPTATION ET TEMPS D'ÉVOLUTION

L'évaluation de la performance est donc un élément clé pour l'adaptation des comités d'experts, et plus largement pour les mécanismes adaptatifs. Dans l'objectif de garantir une évaluation de la performance la plus exacte et fine possible, il est nécessaire de disposer de la « vérité terrain » : le flux de labels. Dans le cas de la fraude bancaire, l'évaluation d'une transaction saine ou frauduleuse peut prendre plusieurs jours voire semaines en fonction du temps d'investigation. La vitesse de récupération des labels peut alors devenir un frein structurel à la vitesse d'adaptation du/des modèle(s).

L'objectif pour les projets mettant en place du monitoring de performance et de la détection de dérive est donc de pouvoir trouver la bonne méthode d'adaptation, afin d'alimenter, de ré-entraîner et de maintenir la performance du/des modèle(s) en production.

RECOMMANDATION	MISE EN OEUVRE	À NE PAS FAIRE
FACILITÉ	Les données doivent arriver de manière fluide et automatique, de même pour le feature engineering.	Demander les données par mail, avec un interlocuteur différent par source de données.
SIMPLICITÉ	L'optimisation d'hyper-paramètres est très chronophage, pour un gain de performance marginal. Une optimisation minimale permet de se concentrer sur les vrais problèmes de mise en production (la robustesse par exemple)	Dépenser 80% de ses ressources informatiques dans la GridSearch sans évaluer le gain de performance par rapport à la configuration par défaut.

RECOMMANDATION	MISE EN OEUVRE	À NE PAS FAIRE
HUMILITÉ	La notion de performance est toujours relative. En plus d'une référence métier, un modèle complexe doit toujours être comparé à un modèle naïf pour s'assurer de la valeur ajoutée. Par exemple, un modèle linéaire avec 5 variables, les plus pertinentes d'un point de vue métier.	Commencer la modélisation par un XGBoost avec 200 variables sans avoir évalué l'apport d'un modèle simple, plus facile à maintenir.
SENSIBILITÉ	Le processus d'apprentissage est un processus aléatoire et possède une variance propre. Elle peut être évaluée par perturbation des données (bootstrap, undersampling) ou du modèle (graine aléatoire). L'évaluation de cette incertitude permet de s'assurer qu'une dérive n'est pas juste due à l'instabilité intrinsèque du modèle.	Discuter des résultats sur la base d'une seule et unique graine aléatoire, sans avoir fait d'étude de sensibilité.
PRUDENCE	Une fois l'incertitude du modèle connue (Cf sensibilité), certaines prédictions seront plus incertaines que d'autres (valeurs manquantes par exemple). Dans ces cas-là, il peut être plus rentable de s'abstenir de prédire plutôt que de donner une prédiction incertaine.	Faire une confiance aveugle dans les prédictions de l'algorithme en toutes circonstances sans mettre en regard les coûts des erreurs et les coûts d'absence de prédiction.
REPRODUCTIBILITÉ	Tous les processus aléatoires de la chaîne de traitement doivent être contrôlés, il ne doit pas y avoir de graine aléatoire cachée. Un test unitaire consiste à lancer deux fois le code et s'assurer que les résultats sont identiques à la précision machine.	Itérer sans s'assurer que les itérations précédentes sont parfaitement reproductibles.

RECOMMANDATION	MISE EN OEUVRE	À NE PAS FAIRE
CAUSALITÉ	En production, le modèle doit toujours prédire sur un jeu de données situé dans le futur du jeu d'entraînement. La seule validation croisée qui soit représentative d'une mise en production est la validation chronologique, où le jeu de test est toujours dans le futur du jeu d'entraînement. De même, le feature engineering doit respecter la causalité de l'information.	Considérer que les performances obtenues par validation croisée sont représentatives des performances de production. C'est nier l'existence des dérives de données.
MODULARITÉ	Le code doit être facilement maintenable, testable, et donc organisé en fonctions élémentaires, simples et courtes.	Lancer des notebooks en série de manière répétitive pour mettre à jour des modèles.

RETOUR D'EXPÉRIENCE

Nous avons réalisé une mission de détection de fraude sur des transactions bancaires. Le problème de dérive était bien connu des services de sécurité : les fraudeurs agissent par vagues, changent constamment de stratégie (sur les montants, sur les commerçants, sur les profils de détenteurs de carte), et ce en l'espace de quelques jours à quelques heures. Nous l'avons d'ailleurs observé très rapidement : un modèle de détection de fraude entraîné à l'instant t devient rapidement obsolète. Nous avons donc mis en place des algorithmes adaptatifs type comités d'experts (forêts aléatoires adaptatives) avec la librairie MOA (*Massive Online Analysis*). Ces algorithmes s'entraînent de manière incrémentale sur un flux de données, ont un système de surveillance de performances intégré, et réinitialisent leurs mémoires automatiquement et de manière optimale quand leur performance fait défaut. De fait, en prenant en compte la dynamique temporelle des fraudeurs, de nombre de fraudes bloquées à raison a sensiblement augmenté sans impacter la gêne client, et le système de priorisation des investigations a vu sa rentabilité croître de 20%.

QUESTIONS À JACOB MONTIEL (TELECOM PARISTECH)

Jacob Montiel est chercheur à Télécom ParisTech et contributeur principal de scikit-multiflow. Ses sujets de recherche touchent à l'apprentissage automatique sur des flux non-stationnaires de données. Il a également dirigé des travaux de développement logiciel pour les moteurs d'avions chez GE Aviation, dans une approche Big Data industrielle.

Qu'est-ce qu'une dérive ?

L'apprentissage est souvent considéré comme une tâche statique. Cependant, en conditions réelles, les données évoluent constamment. C'est ce qu'on appelle une dérive conceptuelle. Par exemple, les marchés financiers sont instables : les risques de crédit ne sont pas les mêmes d'une année sur l'autre.

Quelles conséquences pour les modèles classiques ?

Les modèles standards ont toutes les chances d'échouer, car ils ignorent complètement ce phénomène. Souvent, on développe un modèle et, après l'avoir fait fonctionner une fois, on pense que le travail est terminé. En réalité, il faut toujours évaluer la performance du modèle en production au cours du temps et évaluer s'il est sujet à une dérive.

Quelles sont les solutions ?

Ce phénomène a été largement étudié dans la littérature sur le *stream mining*, où plusieurs solutions ont été proposées. Dans ce cadre, les modèles s'adaptent au fur et à mesure. En effet, certaines méthodes permettent de détecter automatiquement les changements dans les données en surveillant la performance des modèles. Elles permettent d'indiquer : "attention, un changement est en train de s'opérer, il faut réagir maintenant". L'idée est ensuite de créer des modèles qui s'adaptent rapidement aux changements. Par exemple, si une dérive est détectée, on peut rejeter un modèle obsolète et déclencher l'apprentissage d'un nouveau modèle.

Peut-on utiliser ces techniques quand les données arrivent par cohorte ?

Pas directement, car dans ce cas, il faut attendre entre deux acquisitions

successives de données. C'est en effet caractéristique de l'approche statique, car sous l'hypothèse de données identiquement distribuées, il est avantageux d'accumuler autant de données que possible. En *stream mining*, cette phase d'accumulation n'a pas lieu d'être, il n'y a même pas de problématique de stockage car la donnée est exploitée à la volée. De toute façon, pour être réactif et adaptatif, il vaut mieux attendre le moins longtemps possible. Ceci étant dit, des recherches en cours étudient des systèmes hybrides, où une brique statique et une brique adaptative peuvent collaborer.

Quels sont les outils dédiés à l'apprentissage adaptatif ?

Il y a des initiatives comme MOA (Massive Online Analysis), écrit en Java, et scikit-multiflow, écrit en Python. Les deux librairies sont en accès libre et implémentent l'état de l'art des algorithmes adaptatifs en *stream mining*. Elles ont été développées par des chercheurs mais conçues pour les professionnels. Quelques fournisseurs Cloud proposent également des solutions optimisées pour les flux de données, mais la plupart ignorent le problème de la dérive.

L'apprentissage adaptatif est-il crucial pour l'intelligence artificielle en production ?

Dans l'industrie, on ne peut pas toujours reproduire ce qui se fait ailleurs car ce qui fonctionne bien dans un cas peut ne pas fonctionner dans une autre. La clef est vraiment de comprendre en profondeur la donnée, de travailler par itération, d'essayer différentes techniques. La solution est toujours personnalisée et particulière au problème traité. Si la donnée évolue par nature et est produite en masse, alors l'apprentissage adaptatif est en effet très prometteur. Dans ce contexte, il fournit les meilleurs performances et la gestion de ressources informatiques la plus efficace.

Pourquoi y a-t-il un tel écart entre les pratiques universitaires et industrielles ?

Les cycles de développement sont relativement différents. En recherche, il est naturel d'aller toujours plus loin, d'expérimenter et d'échouer. L'industrie est plus prudente, et requiert des solutions non seulement efficaces mais également stables, car les enjeux sont bien plus sensibles. L'industrie utilise ce qui marche dans son contexte et l'adapte à son besoin. Ces deux mondes évoluent dans des directions complémentaires. A n'en pas douter, toute collaboration entre universitaires et industriels est bénéfique pour les deux.

QUESTIONS À NINA BERTRAND (BLECKWEN)

Nina Bertrand est Machine Learning Engineer chez Bleckwen, éditeur de logiciel proposant une solution temps réel de lutte anti-fraude bancaire basée sur du Machine Learning.

Quelles problématiques spécifiques rencontrez-vous concernant le cycle de vie de vos modèles anti-fraude?

Le cycle de vie des modèles est un sujet très intéressant mais complexe. Dans le cas, par exemple, d'un sujet marketing comme le churn, on peut analyser la dérive des données dans le passé et estimer combien de temps un modèle performant peut être laissé en production avant de recommencer à le challenger. Dans le cas de la fraude, le sujet est un cran plus complexe en raison de la nature de ce que l'on essaie de prédire. Les fraudeurs sont réactifs par rapport au système de détection de fraude, toujours plus imaginatifs, et particulièrement motivés par les gains financiers qui sont en jeu. Nous sommes donc particulièrement soumis aux questions de drifts des distributions de données, ce qui nécessite une attention importante à apporter à la performance du modèle et au ré-entrainement régulier.

D'autre part, nous avons aussi des contraintes liées au fait d'être une solution temps réel, ce qui soulève des problématiques particulières dans le cycle de vie du modèle, comme son chargement à chaud afin de ne pas avoir à redémarrer la solution en cas de changement de modèle en production, où comment supprimer un modèle à chaud, l'archiver...

Avez-vous opté pour un réentrainement totalement automatisé, manuel, ou dépendant de règles métier?

Nous avons opté pour un réentrainement à la demande mais totalement automatisé. Un ré-entrainement de modèles implique de nombreuses étapes : ingestion des nouveaux feedbacks/pertes/données client/..., déduplication, anonymisation, nettoyage technique et métier, preprocessing, ajout des différentes catégories de features indispensables à la modélisation de la fraude, mais aussi séparation

de ces données pour l'entraînement / la validation / le test, tuning des paramètres des différents modèles, optimisation des poids du stacking, génération des prédictions et de l'interprétabilité sur certains datasets... Nous avons donc automatisé toutes ces étapes avec stockage intermédiaire des données / pipelines de traitement / différents modèles, de manière à n'avoir qu'à cliquer sur un bouton pour effectuer cette chaîne de traitements (que ce soit en local ou dans le cloud, c'est simplement un paramètre à choisir au moment du run). De cette manière, on obtient les insights nécessaires au Data Scientist pour comparer les challengeurs et les modèles en production, et ainsi déterminer quels modèles sont les plus adaptés aux conditions réelles actuelles.

Le ré-entrainement régulier de manière automatique (plutôt qu'à la demande) est l'une des prochaines étapes envisagées pour notre solution.

Comment prenez-vous en compte le retard dans l'arrivée des labels et/ou l'obsolescence de certains patterns?

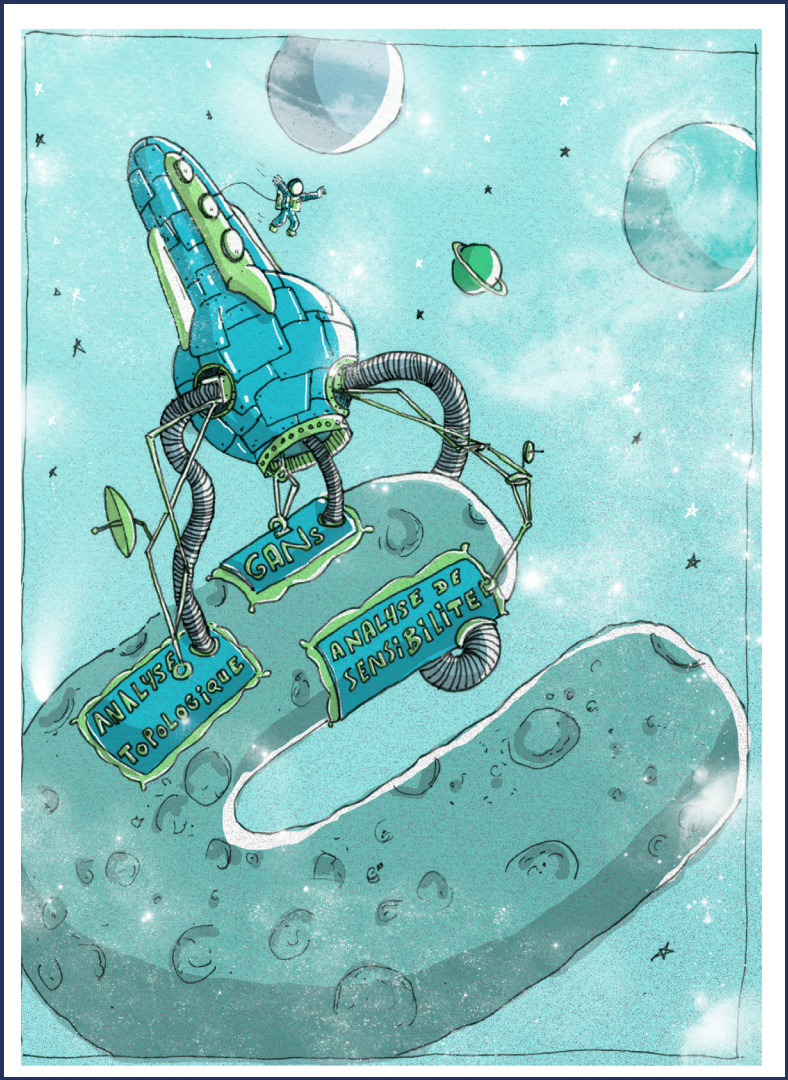
Il y a différentes manières d'adresser le problème de l'obsolescence de certains patterns : on peut utiliser un modèle entraîné sur des données "chaudes", donc entraîné sur des données récentes, et réentrainer ce modèle particulier plus régulièrement en mettant à jour son poids dans le stacking. Mais si le modèle est supervisé, on peut effectivement être bloqué par le retard des feedbacks. Dans ce cas, le mieux est d'intégrer un modèle non supervisé qui s'affranchit des labels et qui se prête bien au sujet de la fraude.

Avez-vous mis en place des outils pour le suivi de la qualité des données / de la performance des modèles / alerter automatiquement en cas de chute de performance ou détection de dérive ?

Effectivement c'est assez indispensable pour mettre un modèle en production. La première étape est de gérer le monitoring, on s'intéresse notamment à la distribution puis au drift des scores, mais aussi à celles des features dans le temps. Les évolutions des différentes métriques de performance dans le temps sont aussi bien sûr des indicateurs à monitorer. L'analyse du monitoring et donc de l'évolution des données et du modèle peut être faite par l'humain, mais cela devient vite chronophage. L'étape suivante est donc logiquement de l'accompagner dans ce suivi grâce à l'alerting : on met en place des alertes basées sur un seuil sur le nombre de transactions "risquées", sur un test statistique vérifiant que la distribution a changé, etc..

5. MESURE ET AMÉLIORATION DE LA ROBUSTESSE : DES OUTILS PHARES POUR FACILITER L'ADOPTION

Contributeurs : *Alberto Guggiola, Nicolas Peltre*



A. INCONNUES CONNUES, INCONNUES INCONNUES

Robustesse, ou robustesses ? En statistiques, la robustesse mesure la capacité d'un estimateur à ne pas être modifié par une petite variation des données ou des paramètres. Mais il ne faut pas négliger la variété d'aspects qui peuvent se dissimuler sous le terme générique de robustesse. Il faudrait en fait toujours spécifier "robuste" par rapport à quoi, et dans quelles conditions. En 2002, le secrétaire à la Défense des Etats-Unis Donald Rumsfeld évoquait un principe valable sans doute en politique, mais avec une application aussi dans le cadre de la recherche scientifique en général, et de l'IA en particulier :

> " There are known knowns [...] We also know there are known unknowns [...] But there are also unknown unknowns "[1]

Bien que nous ayons une connaissance incomplète du monde qui nous entoure, nous cherchons tout de même à effectuer des prédictions grâce à des modèles d'apprentissage automatique. Les valeurs ainsi obtenues sont en général fiables et de bonne qualité, car les patterns se reproduisant sont fréquents (les "connues connues"). Mais ces patterns ne sont pas toujours respectés puisqu'ils dépendent de beaucoup de paramètres, qui sont difficiles à connaître précisément à un instant donné ("inconnues connues"). Pour nuancer l'affirmation d'une prédiction, des méthodes existent depuis longtemps pour **estimer au mieux les incertitudes**. Néanmoins, il est toujours possible d'être confronté à un phénomène absent des données d'entraînement et donc inédit ("inconnue inconnue"). De part sa construction, la robustesse d'un algorithme d'IA est intrinsèquement liée à deux segments complémentaires. D'une part elle **dépend de la nature, de la qualité et de la représentativité des données**, et d'autre part de **la nature même d'un modèle**. Par conséquent, comment peut-on évaluer la robustesse d'un modèle d'IA, et surtout comment peut-on l'augmenter ?

1 "Il existe des connues connues [...] Nous savons aussi qu'il existe des inconnues connues [...] Mais il y a aussi des inconnues inconnues

B. IL Y AVAIT UNE FOIS ... UN MONDE SANS IA

Il est plutôt commun, dans le milieu de la recherche, de “découvrir” les mêmes idées plusieurs fois au sein de communautés différentes ; la robustesse des modèles ne fait pas exception. La conviction de devoir tester la fiabilité d’une prédiction existe depuis longtemps, la comparaison des solutions analytiques théoriques avec les résultats expérimentaux est même la pierre angulaire de la méthode scientifique.

Dès la deuxième moitié du XXème siècle, la possibilité d’obtenir des résultats grâce à des simulations numériques a donné naissance aux premières approches formalisées connues sous le nom de **VVUQ**[2] (pour Verification and Validation, Uncertainty Quantification). Initialement, cette méthodologie s’applique aux modèles numériques déterministes[3], incluant entre autres les implémentations de méthodes de résolution approchée d’équations de dynamique (ex : codes de thermo-hydraulique[4]), de mécanique (ex : résistance des structures[2]) ou encore de biomécanique[5]...

Ces modèles numériques sont généralement multi-physiques, non linéaires, invoquent des échelles d’espace et de temps différentes, et peuvent peu ou prou être considérés comme des boîtes noires. En ce sens que leur structure peut difficilement être appréhendée exhaustivement.

Le parallèle avec des modèles complexes de représentation tels que les réseaux de neurones profonds est criant – bien qu’une différence essentielle réside dans le fait que les paramètres d’entrée (ou caractéristiques/features dans le cadre machine learning) de ces modèles numériques sont généralement des variables aléatoires[6] et non des échantillons de données brutes.

2 Assessing the Reliability of Complex Models: Mathematical and Statistical Foundations of Verification, Validation and Uncertainty Quantification. National Academies Press, Washington D.C., 2012

3 Swiler, L. (2016). VVUQ Best Practices in Computational Science/Engineering Problems with some thoughts about extensions/limits to Complex Systems Models. Sandia Laboratories

4 Review of Uncertainty Methods for Computational Fluid Dynamics Application to Nuclear Reactor Thermal Hydraulics. Nuclear Safety NEA/CSNI/R(2016)4, Nuclear Energy Agency, 2016.

5 Steinman, D.A, Migliavacca, F. (2018). Editorial: Special Issue on Verification, Validation, and Uncertainty Quantification of Cardiovascular Models: Towards Effective VVUQ for Translating Cardiovascular Modelling to Clinical Utility. Cardiovascular Engineering and Technology, 9: 539-543.

6 Ghanem, R., Higdon, D., Owhadi, H. (eds). Handbook of Uncertainty Quantification. Springer, 2017.

L'idée principale des approches VVUQ est de **décorréler trois types d'erreurs conceptuellement différents**, et de les analyser séparément afin de mieux identifier la source du problème et de pouvoir y remédier avec les outils techniques ou méthodologiques adaptés.

Les trois principaux contrôles à réaliser en phase d'audit d'un modèle sont donc :

- **Vérification du code**, qui teste la cohérence entre modélisation mathématique et implémentation computationnelle ;
- **Vérification des calculs**, qui vérifie la cohérence entre le modèle computationnel et les résultats des simulations ;
- **Validation**, qui compare les résultats des simulations avec les observations expérimentales ou les résultats théoriques.

Ces trois volets d'une cohérence globale de l'approche restent d'actualité pour les modèles d'IA, et il convient pour la communauté des Data Scientists et des Data Engineers de le garder à l'esprit. Cependant, la question de la démarche pratique à adopter pour suivre ce cadre reste ouverte.

Dans le domaine de la recherche opérationnelle et de l'optimisation sous contraintes, la robustesse d'une solution est un véritable atout. Le sens du mot "robustesse" est ici à nuancer. Sachant que parmi les paramètres du système, certains sont soit incontrôlables, soit inconnus (respectivement les "inconnues connues" et les "inconnues inconnues"), une solution sous-optimale peut être préférée à une solution optimale mais qui se dégrade rapidement dès qu'un paramètre change. Ce principe est aussi applicable au machine learning, où la solution retenue correspond fréquemment **au meilleur compromis** entre plusieurs aspects parfois en compétition entre eux : la **capacité prédictive**, le **coût computationnel**, la **possibilité d'interpréter** les résultats et, comme on vient de le voir, la **robustesse** par rapport aux changements des entrées ou des paramètres.

C. UN SUJET TELLEMENT INNOVANT... QU'IL EST LA QUESTION #0 POSÉE À UN CANDIDAT DATA SCIENTIST !

Même en restant dans le (de moins en moins) petit monde de l'IA, le sujet est loin d'être nouveau. La contrainte de robustesse des modèles et la nécessité de généralisation du modèle à de nouveaux entrants, tout en évitant le risque de **sur-apprentissage** (i.e. d'apprendre par coeur les données à disposition), ont donné naissance aux principes de la **validation croisée** et de la **régularisation des modèles.**

Un modèle simple présente l'avantage d'être plus généralisable aux données hors de l'ensemble d'entraînement, et donc plus robuste. La robustesse d'un modèle se mesure de manière indirecte en mesurant la qualité de la prédiction de l'algorithme sur des données hors de l'ensemble d'entraînement. Sans rentrer dans le détail de ce sujet, présenté dans tout livre d'introduction au Machine Learning, il est donc légitime de se demander en quoi la robustesse des modèles est encore un sujet de recherche ouvert et passionnant.

D. ATTENTION À NE PAS TOMBER DANS UN TROU CRÉÉ LORS DE L'APPRENTISSAGE !

La robustesse assurée par ces seules démarches reste partielle, car au final elle dépend des données disponibles au moment de l'entraînement du modèle. Lorsque le modèle IA est en production, il doit être en mesure de bien gérer des données qu'il n'aura jamais vues.

Une analyse a priori des nouveaux inputs donnés à un modèle déjà entrainé permettrait d'identifier les cas mal traités, même s'ils n'étaient pas disponibles lors de l'apprentissage. Bien entendu, cette vérification se base elle aussi implicitement sur les données mises à disposition lors de l'entraînement. Pour une nouvelle entrée, nous pouvons par exemple vérifier si elle est proche d'un sous-ensemble du jeu d'entraînement, i.e. vérifier si lors de l'entraînement nous avons eu la possibilité d'entraîner l'algorithme sur des cas similaires ou non (détection d'anomalie dans les observations). En complément, nous pouvons aussi vérifier si

les observations proches du point à étudier (identifiées par exemple grâce à un KNN) correspondent à un label unique. De cette manière, nous pouvons obtenir un score de fiabilité sur les nouvelles prédictions effectuées en production.

Notons, que si la logique de cette vérification est claire, la notion de proximité dans un cas réel est parfois compliquée à utiliser en pratique, notamment à cause du bien connu fléau de la dimension (dans un espace de grande dimension, des données en nombre fini seront en général très éloignées les unes des autres). De plus, cela nécessite de choisir une mesure de la distance, qui est ensuite utilisée pour comparer deux à deux les valeurs d'une même dimension. L'**analyse topologique des données**, discipline mathématique développée afin de quantifier leur structure spatiale, peut nous aider dans ce cadre car elle permet de réduire la dimension du problème, de diminuer l'impact du bruit et de rendre l'analyse indépendante de la métrique choisie [6] [7]. Qui plus est, dans les situations où l'échantillonnage est contrôlable, une telle analyse nous permet d'enrichir notre jeu de données d'entraînement de manière optimale via un plan d'expérience optimisé, afin de réduire le plus possible ces zones d'ombre inconnues du modèle. Cette optimisation peut même être automatisée avec l'**active learning**, afin d'identifier à chaque étape de l'apprentissage la donnée à rajouter afin de maximiser le gain d'information sur le problème et réduire la durée ainsi que le coût d'entraînement [8].

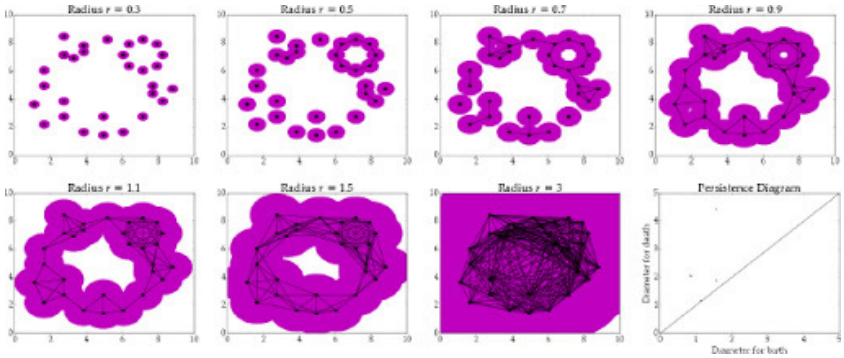


Figure 1 - Analyse topologique : détection de trou
https://learning-analytics.info/journals/index.php/JLA/article/view/5196/6089

E. REMPLISSONS LES TROUS !

Si l'analyse précédente nous révèle des zones d'ombre dans les données à disposition, et acquérir des informations complémentaires est trop complexe ou trop coûteux, il existe une manière de contourner le problème : l'**augmentation de données**, très utilisée notamment dans le domaine de l'analyse d'images. Cette stratégie consiste à créer des nouvelles données en effectuant des modifications mineures (telles que des rotations ou des translations) à celles que nous avons, de manière à donner un ensemble d'inputs plus complet à nos algorithmes.

Une croyance répandue veut qu'un nombre de données élevé implique une meilleure performance de l'algorithme entraîné. Cependant, même si nous disposons de téraoctets de données homogènes, l'algorithme ainsi créé risque de ne pas être aussi performant lorsqu'il sera en production. En effet, dans un cas d'usage de type analyse d'images, l'algorithme devra faire face à un spectre étendu de conditions hétérogènes en termes d'orientation, positionnement, taille, luminosité des images. L'augmentation de la donnée permet alors de créer artificiellement de telles conditions en augmentant la variété des données à disposition, et donc *in fine* de rendre l'algorithme plus robuste. [9]

F. LA SENSIBILITÉ : UN SUJET... SENSIBLE

Tout comme la définition statistique de la robustesse, lorsqu'un modèle est robuste nous nous attendons à ce que des données similaires engendrent des réponses similaires.

Cette intuition est bien ancrée en nous probablement à cause de la multiplicité des systèmes simples autour de nous qui ont des propriétés de linéarité : si j'augmente légèrement l'effort lors de ma session de jogging, je ne m'attends pas à passer à une vitesse de 100 km/h.
Mais il faut parfois savoir se méfier de sa propre intuition. La présence de phénomènes de seuil est récurrente dans les systèmes complexes. Par exemple une augmentation du trafic quasi-imperceptible peut générer un embouteillage sur l'autoroute (ou comme le veut la célèbre métaphore introduite par E. Lorentz "Le battement d'ailes d'un papillon au Brésil peut-il provoquer une tornade au

Texas ?"). Dans le domaine de l'optimisation, un petit changement dans les paramètres comme une contrainte plus stricte à satisfaire peut transformer un problème trivial en un problème non calculable avec une complexité raisonnable. De manière encore plus évidente, dans un cas de classification il y aura toujours, par construction, des frontières entre des points assignés à différents labels i.e. qui appartiennent à différents bassins d'attraction, et dans ces régions un petit changement des entrées provoquera un grand changement dans la sortie.
Même un excellent modèle peut donc être très sensible à certains changements dans les entrées. En revanche une dépendance très forte de la sortie à une variable qui ne devrait pas être très impactante est un possible signal de sur-apprentissage (ou de mauvaise compréhension de la mécanique d'un point de vue métier).

Dans l'**analyse de la sensibilité** on retrouve les deux possibles angles de lecture qui existent dans le domaine de l'interprétation des modèles : le **niveau local et global.**
La sensibilité globale vise à quantifier comment la variabilité des entrées se répercute sur celle de la sortie, en déterminant quelle part de variance de la sortie est due à tel ou tel ensemble de variables. L'une des mesures de la sensibilité s'effectue via les **indices de Sobol**, et notamment l'indice de Sobol de premier ordre. Cet indice est défini pour chacunes des variables et permet de quantifier la part de variance de la sortie due à la variable étudiée. Pour cela, on calcule à quel point la variance de la sortie décroît si on fixe la variable étudiée. Mathématiquement, cela revient à calculer la variance de l'espérance conditionnelle de la sortie associée à la variable étudiée, normalisée par la variance de la sortie. Une généralisation intéressante (indices de Sobol totaux) prend en considération également les interactions entre la variable étudiée et toutes les autres variables. Cet indice exprime la sensibilité totale de la variance de la sortie à la variable étudiée sous toutes ses formes, c'est-à-dire en prenant en compte les interactions de la variable étudiée avec les autres variables et les éventuelles dépendances. [10]

Comme dans le cadre de l'intelligibilité, ce niveau macroscopique nous permet d'avoir une première vision de la situation, mais est trop agrégé pour permettre une compréhension fine. Pour ce deuxième besoin, les analyses de sensibilité locale sont plus adaptées : l'impact d'une petite variation autour d'une dimension donnée pour une entrée est étudié. Dans le cas récurrent de modèle coûteux en

temps d'exécution, les méthodes dites "*One At a Time*" (OAT), dont la **méthode de Morris** fait partie, sont privilégiées pour entre autres effectuer un premier screening des variables d'entrées influentes.

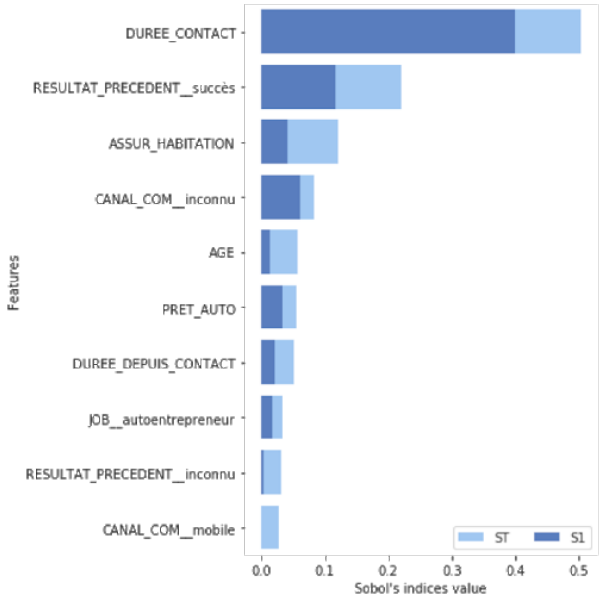


Figure 2 - Exemple de calcul d'indices de Sobol de premier ordre et totaux sur le dataset car_insurance

G. DÉRIVE DU MODÈLE OU ANALYSE DE ROBUSTESSE ? LES DEUX EN MÊME TEMPS !

En parlant de robustesse, deux dimensions principales sont envisageables derrière ce terme. D'une part, les méthodes précédemment évoquées nous apportent plus de transparence vis-à-vis de notre confiance dans le comportement du modèle entraîné. Nous savons dès lors quelles sont les typologies de données pour lesquelles nous considérons l'algorithme comme fiable, et quel peut être l'impact d'un changement dans les entrées ou dans les paramètres grâce aux méthodes de sensibilité.

D'autre part, les stratégies présentées dans le chapitre 4 permettent de suivre les performances du modèle en production, cela afin d'identifier de possibles dérives de performance et décider du moment le plus adapté pour ré-entraîner les algorithmes.

Ces deux stratégies ne sont pas à considérer comme mutuellement exclusives : suivre l'évolution de la robustesse d'un modèle en production (et non juste ses métriques métier) peut nous permettre de repérer une baisse de fiabilité. Supposons qu'à performance égale, une variable jusqu'à présent peu significative devienne de plus en plus importante dans le modèle. Cela peut être le signe d'un changement dans le phénomène modélisé, tel qu'un changement dans la structure des données ou une évolution réglementaire, qui pourrait engendrer par la suite une dégradation des performances si cela n'est pas géré.

Dans le cas d'un score d'appétence, nous pourrions observer une augmentation de la sensibilité à la région géographique du prospect. Après analyse, nous pourrions découvrir qu'un nouveau concurrent vient d'arriver dans une zone, expliquant alors ce changement de paradigme. Il faut alors effectuer une mise à jour des modèles, avant même de mesurer une dégradation conséquente des KPIs métier.

H. RÉSISTER AUX ATTAQUES : L'IMPORTANCE DE L'*ADVERSARIAL LEARNING*

Jusqu'ici, nous avons discuté de la robustesse par rapport à des événements qui peuvent être classifiés comme "aléatoires". **Dans d'autres situations, au contraire, des acteurs ont intérêt à identifier et exploiter les faiblesses du système.** Nous revenons donc à une définition peut-être plus intuitive de la robustesse : comment le modèle va-t-il réagir face à une attaque ? Encore une fois, il s'agit de considérations nées bien avant l'intelligence artificielle, qui ont été ensuite projetées dans ce paradigme : au moment de déployer n'importe quelle méthode de lutte anti-fraude (algorithmique ou basée sur des règles métier) nous devons prendre en considération le fait que les fraudeurs vont rapidement adapter leurs stratégies afin d'éviter les comportements désormais "interdits" et de les substituer avec d'autres, qui ne seront pas bloqués.

Une démarche standard pour vérifier la robustesse d'un système est d'effectuer un stress test. Ceci consiste à simuler les actions d'un acteur mal intentionné et d'en mesurer l'impact. En fonction des résultats, nous pouvons repousser la mise en production si les risques sont encore trop élevés (modèle pas assez "robuste"), ou alors prioriser une liste de contre-mesures pour pallier les faiblesses identifiées. Dans tous les cas, il s'agit d'une démarche itérative car les adversaires vont s'adapter aux nouvelles contre-mesures (par exemple dans un problème de fraude quelconque).

Les projets d'intelligence artificielle risquent d'être particulièrement sensibles à ce sujet. Les algorithmes d'apprentissage automatique ne sont pas, de manière générale, construits pour réagir à ces comportements intelligents et adaptatifs. Même une approche capable de garantir des performances excellentes pourra garder des points de vulnérabilité spécifiques, activables via une manipulation ad hoc des données d'entrée : dans un grand nombre de cas d'usage (reconnaissance biométrique, sécurité informatique, score de crédit) ce phénomène risque à lui tout seul d'empêcher l'applicabilité de méthodes analytiques plus avancées. L'ampleur du possible impact médiatique négatif a été déjà prouvée avec les *adversarial examples*, des images bien classifiées par un réseaux de neurones sont très légèrement perturbées de manière à changer la prédiction du modèle pour une erreur avec un taux de confiance pourtant très élevé. Des papiers récents ont montré que des petites perturbations physiques, simulant des graffitis, à un panneau de stop peuvent amener un réseau de neurones profond à le classifier comme une limitation de vitesse. Juste à titre d'exemple, nous pouvons imaginer l'impact que des tels points de faiblesse pourraient avoir sur l'adoption à grande échelle des voitures autonomes …

La bonne nouvelle, c'est que l'intelligence artificielle nous met à disposition des nouveaux outils pour mener ce combat. Les *réseaux antagonistes génératifs* (GAN en anglais) se composent de deux réseaux de neurones qui vont s'affronter et s'améliorer au fur et à mesure.

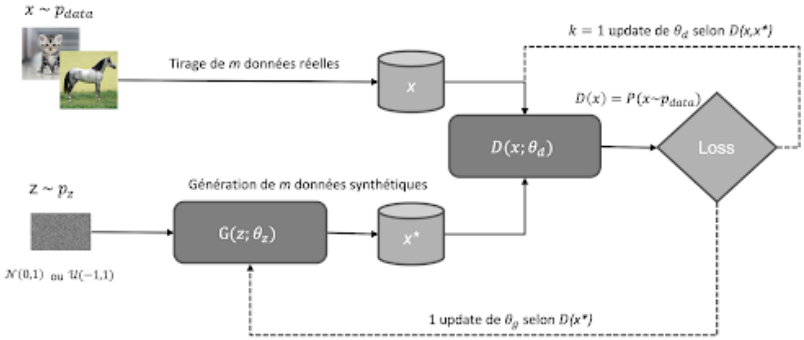


Figure 3 - Architecture d'un GAN

Le premier réseau de neurones générateur est entraîné pour générer des données à partir de bruit. La donnée est ensuite transmise au réseau de neurones discriminant qui a pour objectif de discerner les données réelles des données simulées. La prédiction du réseau discriminant est ensuite rétro-propagée aux réseaux générateurs et discriminateurs afin que ceux-ci affinent leurs sorties. Le réseau générateur peut ainsi construire des exemples de plus en plus réalistes. **La même logique peut dans le principe être adaptée afin de générer des exemples antagonistes**, i.e. qui seront probablement mal gérés par le système discriminateur D, et utiliser ces inputs "difficiles" pour augmenter la robustesse de notre modèle. [11]

Ce type de réseau est très utilisé dans les modèles biométriques, où une très faible modification de l'image peut provoquer une mauvaise classification [12]. Les réseaux GAN trouvent également des applications prometteuses, notamment dans le screening de molécule dans l'oncologie [13].

1. la gestion des risques associés aux modèles
2. un système d'entités décisionnelles
3. protection des données
4. se fier à quelqu'un ou à quelque chose
5. processus d'amélioration perpétuelle

6. VERS LA MISE EN PLACE D'UNE GOUVERNANCE DES MODÈLES

***Contributeurs** : Florian Canderlé, Jonathan Cassaigne*

Au delà de la gouvernance des données, pourquoi et comment mettre en place une gouvernance des modèles ?

La nécessité de gouverner les données de l'entreprise, de les gérer comme des actifs importants, est désormais bien établie dans la quasi totalité des organisations.

La prise de conscience que le cycle de vie d'un modèle data ne s'arrête pas à sa mise en production et la multiplication de ces modèles au sein des processus métier clés des organisations font maintenant émerger un besoin grandissant de gouvernance plus globale des modèles data.

En effet, ils ont vocation à devenir des assets de plus en plus décisifs pour les organisations et comme tout patrimoine de l'entreprises, ils doivent être managés, gouvernés et sécurisés.

Si cette vision de la gouvernance des modèles est plus ou moins évidente dans des secteurs comme la finance où la gestion des risques induits par les modèles data est déjà largement réglementée, elle reste encore vierge dans la majorité des autres secteurs. Il est encore courant de voir les équipes IT des entreprises gérer les modèles IA comme des applications d'entreprise "classiques". Or, le cycle de vie des modèles IA étant notamment dicté par la vitesse des changements dans les données, ils nécessitent plus de vitesse, d'agilité et de réactivité et imposent de nouvelles exigences.

De plus les problématiques organisationnelles et de gouvernance sont souvent les premiers freins à la bonne gestion du cycle de vie des modèles. En effet si le processus projet et les responsabilités entre les acteurs Métier, Data et IT sont désormais généralement bien établis pour la construction d'un modèle algorithmique, cela n'est que rarement le cas pour les modèles déjà en production. Par exemple, si nous retrouvons des modèles organisationnels différents selon les entreprises, il est toutefois courant que l'équipe IT soit en charge du maintien en condition des applications de l'entreprise. Une des problématiques rencontrées est alors la disponibilité des Data Scientists, regroupés au sein d'une équipe data dont l'objectif est le développement de nouveaux projets, lorsqu'il faut intervenir sur ces modèles en production.

Tout cela nécessite un investissement en temps et en ressources non négligeable, et donc la mise en place d'une gouvernance adaptée. Il n'y a encore que peu de littérature sur la gouvernance globale des modèles en production et peu d'entreprises l'ont réellement mise en place.

Pour mieux comprendre les enjeux de la gouvernance des modèles, nous nous sommes intéressés dans un premier temps au cas du secteur financier. Il est en effet pertinent de comprendre si le *Model Risk Management* mis en place à des fins initiales de respect de la réglementation est adapté aux nouveaux modèles d'IA et transposable à d'autres secteurs.

C'est sur cette base que nous tâcherons ensuite de définir les grands objectifs, principes et leviers d'une gouvernance des modèles adaptée à toutes les entreprises et secteurs d'activité.

A. FOCUS SUR LE MODEL RISK MANAGEMENT MIS EN PLACE DANS LE SECTEUR FINANCIER : EST-IL ADAPTÉ AUX MODÈLES IA ET EST-IL UN EXEMPLE DE GOUVERNANCE DES MODÈLES POUR LES AUTRES SECTEURS ?

Le nombre de modèles qu'utilisent les banques croît de manière exponentielle : entre 10 et 25% d'augmentation par an[1]. Ils interviennent dans toute la chaîne de valeur des banques, que ce soit dans leurs processus internes afin d'améliorer leur efficacité (origination, gestion des limites, recouvrements...), dans leur relation client pour fidéliser et engager (simulations temps réel, campagnes marketing...), ou au niveau réglementaire (fraude, lutte anti-blanchiment, financement du terrorisme...). Ces modèles intègrent de plus en plus de données et deviennent de plus en plus complexes, intégrant notamment des briques d'intelligence artificielle. Cela a poussé les régulateurs à intervenir afin de minimiser les risques liés à la mauvaise utilisation ou mauvaise compréhension des modèles et de garantir la solidité financière des établissements financiers.

Les banques ont donc mis en place une fonction de gestion du risque des modèles, la fonction MRM. Nous allons essayer de comprendre si certains des éléments appliqués dans ce contexte seraient transposables à la gestion du cycle de vie de modèles data dans d'autres secteurs, en particulier pour des entreprises moins matures dans cet exercice.

HISTORIQUE

La notion de Model Risk Management (MRM) est apparue au début des années 2000 suite à un communiqué rédigé par l'OCC, un organe du Département du Trésor Américain. L'objectif était de fournir les lignes directrices permettant de se prémunir des risques associés à la validation, aux tests et à l'utilisation de l'informatique pour les modèles financiers. Suite à la crise de 2008, le Comité de Bâle (BCBS) et l'Autorité Bancaire Européenne (EBA) ont définit les lignes directrices de supervision des modèles internes de risque de crédit.

Depuis, la réglementation a évolué et s'est élargie à l'ensemble des modèles algorithmiques et notamment à tous les modèles prédictifs. Le secteur financier

1 source:
https://www.mckinsey.com/business-functions/risk/our-insights/the-evolution-of-model-risk-management

s'est enrichi en guides de bonnes pratiques de modélisation comme par exemple le guide TRIM en 2017 (Targeted Review of Internal Models) permettant d'assurer que les modèles utilisés sont conformes avec la réglementation. Le respect de la réglementation nécessite aujourd'hui la mise en place d'une fonction spécifique dédiée en interne : la fonction MRM, et donc d'une organisation, de processus et d'outils ad hoc.

OBJECTIFS

Les régulateurs imposent plusieurs niveaux de monitoring de gestion des risques des modèles :

- Ils doivent être compris par le Board et les équipe de senior management des institutions financières. Cela oblige les équipes data à développer des modèles compréhensibles et donc bien documentés puis d'informer l'ensemble du management de l'entreprise sur le fonctionnement du modèle.
- Ils doivent être suivis et revus de manière "continue". Chaque institution financière doit donc mettre en place la politique de monitoring ad hoc.
- Ils doivent être "reproductibles". C'est à dire que les institutions financières doivent garantir la possible "comparabilité des modèles et la réduction de la variabilité entre les différents acteurs".

La mise en place du MRM a donc été pensée pour permettre d'améliorer les bénéfices d'une organisation selon trois axes :

- La réduction des coûts par l'amélioration de l'efficacité et des processus, notamment sur le développement et la validation des modèles
- L'évitement des pertes par l'élimination des modèles inefficaces, peu qualitatifs ou risqués
- L'amélioration du capital par la mutualisation des modèles et la création d'une vision globale transverse

Ainsi, le MRM permet d'une part de réduire la volatilité du P&L en réduisant les risques et en gérant mieux leurs impacts, et d'autre part de mieux piloter les investissements et les priorités business. Cela contribue également à une meilleure transparence et à une culture de gestion des risques. Ces objectifs sont globalement ceux qu'on associe à la gouvernance des modèles.

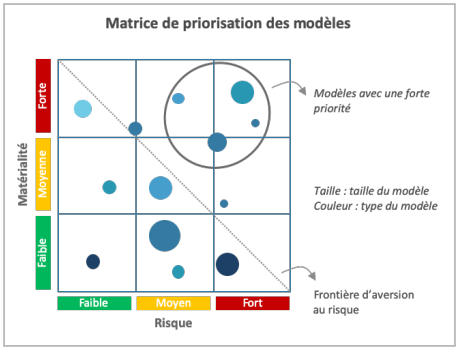
MÉTHODOLOGIE ET ORGANISATION

Le MRM est intégré à l'ensemble du cycle de vie des modèles. Il faut pouvoir anticiper dès les phases de conception et de validation les besoins de monitoring et de reporting des modèles.

La méthodologie consiste à prioriser l'ensemble des modèles afin de sélectionner pour chacun les mesures adaptées. Pour cela, il convient de recenser les différents modèles avec une vision globale et transverse à l'entreprise : quels modèles, pour quels usages, quels utilisateurs et responsables, quels impacts métier et quelle sévérité des risques associés ?

Ces questions permettent dans un second temps de les analyser de manière qualitative et quantitative, en suivant une approche de scorecard de model risk assessment.

Cela donne lieu à une priorisation qui s'appuie sur une matrice matérialité / risque. La matérialité signifie l'impact qu'a le modèle sur l'entreprise, ses activités et sur ses capacités à générer de la valeur : volumes financiers en jeu, contexte d'utilisation (impact réglementaire, lien avec d'autres modèles critiques, prise de décision automatique) et impact financier en cas de défaillance.



Le risque peut se traduire par plusieurs éléments : la qualité et l'intégrité des données en entrée, les hypothèses de modélisation (maturité du modèle / utilisation, complexité, obsolescence), les aléas (comportement dans des conditions extrêmes, sensibilité aux variations, évolution du contexte d'utilisation, stabilité et robustesse), la précision (performance réelle vs théorique, limites de validation), l'environnement d'utilisation (infrastructures SI, gouvernance et contrôles, l'utilisation hors du périmètre initial, dépendances vers d'autres modèles).

Suite à cette priorisation, des instances de revue et de contrôle des modèles sont mises en place en fonction de la priorité de chaque modèle, de manière régulière

afin de mettre à jour l'analyse de risque et de prendre les actions nécessaires.

Une organisation et une gouvernance adéquate est mise en place afin de gérer l'ensemble de ce processus, généralement rattachée à la direction des risques (cf. interview BPCE). Celle-ci se décompose couramment suivant trois lignes de défense : une première ligne en charge du développement et de l'utilisation du modèle, une seconde ligne en charge de la validation et du contrôle, et une troisième ligne en charge de l'audit.

Nous pouvons observer trois étapes afin de mettre en place de la fonction MRM au sein des institutions financières :

1. La mise en place des fondations : processus de revue et de validation (avec rôles et responsabilités), inventaire des modèles (sans standards), début d'équipe de gouvernance et de validation

2. L'implémentation des fonctions clés : vision globale des risques, processus de contrôle automatisés, quantification des risques via des scorecards, formation des parties prenantes (profils seniors), mise en place d'outils

3. La captation de valeur : vision stratégique et priorisation des modèles en fonction de leur impact business, mise en place d'un centre d'excellence pour le développement de modèles, optimisation des ressources, reporting intelligent

CONCLUSION : UNE SOURCE D'INSPIRATION À ADAPTER

Le MRM, de part sa fonction de surveillance des modèles, présente des éléments transposables dans un contexte de gestion des modèles data pour tous les secteurs. Il y a cependant quelques différences notables :

- Des cycles de revue et de vérification différents : le focus historique sur les problématiques de réglementation (annuel ou trimestriel selon les priorités) n'est pas toujours approprié, par exemple dans la vente au détail où une dérive peut se traduire par un impact immédiat sur le CA

- Des typologies de modèles différents (par exemple sans mise à jour des données ou des features en temps réel) : si le MRM vise en cible à couvrir l'ensemble des modèles, dans les faits, ce n'est pas encore toujours le cas

› Une organisation et des processus en place pour réaliser les validations afin de sécuriser les aspects réglementaires : le secteur financier a un coup d'avance sur la mise en place de la gouvernance des données, des départements qualité et des processus de validation des modèles

Enfin, l'estimation du risque des modèles basée sur du machine learning représente une piste intéressante pour les organisations, car la complexité accrue de l'IA a une incidence directe sur les risques associés. Il est par exemple plus difficile de déterminer la fréquence appropriée de ré-étalonnage et de détection des erreurs d'exécution lorsque les modèles changent dynamiquement à un rythme élevé.

B. AU DELÀ DU MRM, PRINCIPES ET LEVIERS DE LA GOUVERNANCE DES MODÈLES

Comme nous venons de le voir, la gouvernance des modèles data existe déjà partiellement dans le secteur financier, principalement grâce aux contraintes réglementaires toujours plus importantes. Cependant, la majorité des autres secteurs sont moins matures et ne sont donc pas encore arrivés au stade d'en ressentir le besoin.

Nous pouvons faire un parallèle intéressant avec la gouvernance des données qui, souvent délaissée dans les phases de POC IA, se révèle aujourd'hui cruciale pour le passage à l'échelle et l'industrialisation en masse des projets data (voir les bonnes pratiques indiquées dans le chapitre 1). Avec la multiplication des modèles IA dans tous les secteurs, il y a donc fort à parier que ce besoin va croître et que les entreprises vont devoir se doter d'organisations adéquates.

Cette section a donc vocation à définir les grands enjeux, bénéfices et principes d'une gouvernance adaptée à tous.

OBJECTIFS DE LA GOUVERNANCE DES MODÈLES

Elle vise à garantir l'efficacité, la sécurité et la conformité des modèles data en production utilisés par une organisation.

Sa mise permet d'atteindre plusieurs objectifs :

› Automatisation et industrialisation de la gestion du risque des modèles : il s'agit ici de définir les conditions de validation d'un modèle, les principes de monitoring du risque et de correction des dérives, en adéquation avec les méthodologies citées dans les chapitres précédents.

› Gain de temps dans la production des modèles en capitalisant sur les modèles existants, par exemple en maintenant le code source, en associant le modèle avec ses fichiers de formation ou en archivant les résultats des essais

› Respect de la réglementation dans le cas de certains secteurs, comme celui de la finance ou prochainement celui de la santé (réglementations aujourd'hui peu matures mais qui devraient voir le jour dans les prochaines années) qui imposent un contrôle fort des modèles. Certaines réglementations sectorielles peuvent nécessiter de s'assurer que les modèles data peuvent générer des résultats reproductibles, traçables et vérifiables.

LES PILIERS DE LA GOUVERNANCE DES MODÈLES

Tout comme la gouvernance des données, la gouvernance des modèles s'appuie sur 4 piliers que nous détaillons ci-dessous:

› Rôles et responsabilités associés au cycle de vie du modèle en production;

› Processus cibles à mettre en place pour garantir efficacité, conformité et sécurité des modèles data;

› Outils dédiés à la gouvernance des modèles;

› Standards et normes à partager au sein de l'organisation.

RÔLES ET RESPONSABILITÉS

La gouvernance des modèles est généralement confiée aux DSI qui sont déjà en charge de la gouvernance de l'ensemble du patrimoine et des applications IT de l'entreprise. Néanmoins cette gouvernance doit absolument être transverse aux organisations et mobiliser en plus de l'IT, l'entité Data et l'entité Métier exploitant le modèle en question.

Nous pouvons regrouper les rôles et responsabilités en trois niveaux. Le premier est centralisé (Model Risk Officer + Model Steward), il met en place la gouvernance au niveau du groupe et est en charge de la vision transverse et du suivi des

risques. Il est généralement confié à la DSI. Le second est à la maille projet, avec la définition des besoins, le développement, l'implémentation et l'utilisation du modèle (Model Owner + Developer + Implementer + User). Enfin, le troisième est en charge du suivi et de la validation des modèles (Model Validator).

Nous avons détaillé ci-dessous les principales responsabilités de ces différents rôles :

Model Governance Officer et Model Steward - *souvent la DSI en coordination avec l'équipe data :*

- Définit les méthodologies et les standards (règles de développement, framework de validation, directives de documentation, mise en place d'outils, etc.)
- Gère l'inventaire et centralise / documente les modèles
- Pilote la priorisation, la revue des risques, le plan de mitigation et le reporting

Model Owner et Model User - *Équipe Métier concernée par l'utilisation du modèle :*

- Définit l'objectif du modèle par rapport aux besoins / enjeux business
- S'assure de la bonne utilisation du modèle dans un contexte donné
- Est responsable des décisions prises par le modèle
- Est responsable des évolutions du modèle
- Utilise le modèle au quotidien
- Donne des feedbacks par rapport au besoin

Model Developer et Implémenter - *Equipe data (data science et data engineer)*

- Développe le modèle par rapport aux besoins
- Est responsable des hypothèses et des limitations techniques
- Test le modèle en pré-production puis en production par rapport au besoin initial
- Implémente le modèle
- S'assure de la qualité du code
- Implémente les évolutions du modèle

Model Validator - *Au sein de la DSI ou de l'équipe data selon les organisations*

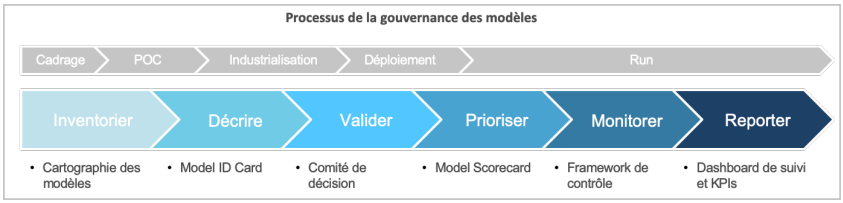
- Vérifie la validité conceptuelle du modèle (revue des limitations, stress testing, analyses de sensibilité) en fonction des politiques globales définies par le Model Risk Officer
- Teste la qualité des données (internes et externes) en coordination avec la gouvernance des données et vérifie la qualité du code
- Vérifie l'applicabilité des conditions business / de marché
- Suit la performance dans le temps (précision, stabilité) et la compare à l'initiale
- Quantifie et mitige le risque associé (en lien avec le Model Owner)

Pour mettre en place cette organisation, il y a évidemment une phase de montée en compétences et en maturité des différents acteurs à prévoir.

On commencera d'abord par la nomination d'un Model Governance Officer et la définition des grands principes de gouvernance associés avant de déployer l'ensemble de ces rôles.

PROCESSUS

La gouvernance des modèles suit un processus qui doit se dérouler en parallèle des projets data, avec des interactions régulières et non dans une tour d'ivoire sans lien avec la réalité des métiers.



La première étape est de recenser les différents modèles avec une vision transversale. Recenser les modèles qui n'ont pas été industrialisés (manque de données, précision non suffisante, etc.) est une bonne pratique car elle permet de capitaliser. Cela permet de construire une cartographie des modèles qui a plusieurs finalités : la première est d'avoir une vision globale et transverse des

modèles et des risques associés, la seconde est de pouvoir capitaliser sur ce qui a déjà été fait, afin de partager les bonnes pratiques et la troisième est de piloter les investissements et les efforts en R&D.

La seconde étape est de s'intéresser à chaque modèle afin de détailler :

- Les objectifs et finalités du modèle
- Les usages et le contexte d'utilisation
- Les concepteurs et utilisateurs
- Les données en entrée / sortie en lien avec la gouvernance des données
- Les risques associés

Tel qu'évoqué dans les chapitres précédents, il est important de définir les processus qui assureront la traçabilité du modèle et donc la capacité de l'organisation à venir le modifier si nécessaire. Pour assurer la traçabilité, les données originales utilisées par les modèles et les scripts d'ingénierie des données utilisés pour transformer et enrichir les données doivent être archivés, ce qui permet de visualiser toutes les modifications apportées aux données tout au long du cycle de livraison.

De plus, pour assurer la vérifiabilité, tous les résultats doivent être archivés, ce qui permet de prouver qu'un modèle fonctionne conformément aux spécifications.

Avant d'être déployé, le modèle doit être validé dans les instances de décision adéquates afin de vérifier qu'il répond à la problématique métier, que l'architecture technique est bien définie, que les risques sont bien identifiés, etc.

Dans un quatrième temps, il s'agit de scorer chacun de ces modèles suivant différents critères afin de les prioriser. Pour cela, il peut être intéressant de s'appuyer sur la méthodologie MRM du secteur financier, à adapter : chaque entreprise doit définir sa matérialité et ses risques en fonction de son contexte et de ses enjeux business.

Suite à cette priorisation, les modèles peuvent être regroupés par famille, avec un processus de surveillance et de validation adapté à chaque famille : les modèles les plus critiques seront surveillés régulièrement (voire automatiquement), avec un

processus de gestion de crise bien défini alors que des modèles moins critiques pourront être monitoré en mode “best effort”.

Enfin, la dernière étape est le reporting, avec le suivi d’indicateurs de pilotage.

OUTILS

Si de nombreux outils ont été évoqués dans les chapitres précédents, nous allons nous focaliser dans cette partie sur ceux dédiés à la gouvernance des modèles. Plusieurs outils commencent à être déployés dans les entreprises les plus matures. En effet, de la même façon que les entreprises traitent depuis quelques années déjà les données comme des actifs, à mesure que le *Machine Learning* devient plus central au sein des opérations, les modèles IA sont traités comme des actifs importants.

Ces outils permettent d’avoir une bonne vision de la cartographie des modèles de l’entreprise et des interdépendances entre chaque modèle. Cela doit permettre à l’entreprise d’éviter de refaire des modèles similaires (ex: 50 modèles de forecast similaires dans différentes entités S&OP d’une grande entreprise), de sécuriser et protéger les modèles construits ou adaptés pour des applications spécifiques de l’entreprise.

Bien que la saisie des métriques et des métadonnées soit tout à fait faisable dans un simple tableau Excel, un environnement data mature peut contenir des dizaines de modèles et de ressources associées. Il n’est alors pas évident de présenter simplement ces informations dans un format facile à utiliser.

Un bon outil de gouvernance de modèle doit être le meilleur poste d’observation des modèles possible. Les grandes fonctionnalités de ces outils sont généralement les suivantes :

- Gestion des autorisations et la sécurité des modèles : qui a accès en lecture et/ou en écriture à certains modèles ?
- Catalogue ou base de données qui répertorie les modèles, y compris la date à laquelle ils ont été testés, formés et déployés. La précision des différents modèles est également indiquée.
- Gestion du versioning des modèles, des fonctionnalités et des données.

Plusieurs outils offrent la possibilité de déployer, de revenir en arrière ou d'avoir plusieurs versions live.

› Gestion des métadonnées et artefacts nécessaires pour une piste de vérification complète.

› Gestion des rôles et responsabilités : qui a approuvé et mis le modèle en production, qui est en mesure de surveiller son rendement et de recevoir des alertes et qui en est responsable.

› Tableau de bord fournissant des vues personnalisées pour toutes les parties prenantes (opérations, ingénieurs ML, data scientists, propriétaires d'entreprises).

Exemple d'outils intégrant plusieurs fonctionnalités de gouvernance des modèles

STANDARDS ET NORMES

La fonction des standards et des normes va être de définir ce qu'est un modèle de qualité et conforme à la politique de l'entreprise ainsi qu'un modèle présentant des risques plus ou moins forts pour l'organisation.

Par exemple, en commençant par le début, une norme de base pourra être de définir que tous les codes sources et versions des modèles doivent être capturés à l'aide d'un outil de contrôle du code source (par exemple, GitHub ou GitLab) permettant aux data scientists de collaborer pour développer rapidement des modèles mais aussi de conserver les éléments clés des modèles.

Autre exemple de norme utile pour assurer la reproductibilité des modèles : l'association entre le code source d'un modèle et les données qui ont été utilisées pour sa construction doit être archivée dans un outil central.

Les organisations ayant déjà implémenté un processus *DataOps* correctement instrumenté ont généralement déjà une réponse pertinente à la définition de ces standards et normes notamment pour ce qui concerne les exigences de qualité de code et la documentation associée.

C. DE LA GOUVERNANCE DES DONNÉES À LA GOUVERNANCE DES MODÈLES, UNE CONDUITE DU CHANGEMENT FORTE POUR L'ADOPTION ET LA BONNE UTILISATION DES MODÈLES AU SEIN DES ORGANISATIONS

Les bonnes pratiques de prise en compte du cycle de vie complet des modèles data et la mise en place d'une gouvernance des modèles permettent donc de minimiser les risques de dérive et de maximiser la "réussite technique" des projets.

Si, pour le moment, peu d'entreprises ont réellement formalisé une gouvernance complète des modèles, cette tendance devrait se développer avec quelques années de décalage sur le déploiement des politiques de gouvernances de la données.

	GOUVERNANCE DES DONNÉES	GOUVERNANCE DES MODÈLES
OBJECTIFS	Favoriser l'accès et l'exploitabilité à une donnée définie, de qualité et de façon sécurisée et éthique	Garantir l'efficacité, la sécurité et la conformité des modèles data en production
BÉNÉFICES	• Accélérer les projets data • Diffuser la culture data • Respecter la réglementation (RGPD)	• Gérer le risque de dérive et de mauvaise utilisation des modèles • Rationaliser et accélérer les modèles existants • Respecter la réglementation (selon les secteurs)
PILIERS	1. Rôles et responsabilités : Data Owner, Data Stewart, Data manager, etc. 1. Processus de gestion de la donnée 1. Outils: Colibra, Informatica, Alation, etc. 1. Normes et standards d'une donnée de qualité, exploitable et partageable	1. Rôles et responsabilité: Model Owner, Model developer, Model validator, etc. 1. Processus de gestion des modèles 1. Outils: MLflow, Datatron, John Snow 1. Normes et standards d'un modèle apte à être mis en production

Illustration de la complémentarité des approches de gouvernance des données et des modèles

Les raisons du développement de la gouvernance des modèles sont multiples. Tout d'abord, pour toutes les industries très régulées, comme le secteur financier ou la santé, cela devient obligatoire de monitorer les modèles en production et de mettre en place l'organisation et la gouvernance nécessaires pour éviter tout risque de dérive.

Ensuite, la multiplication des modèles dans les grandes entreprises - modèles pouvant être développés par des équipes data différentes ayant chacune leur propre méthodologie de travail - va obliger les organisations à définir un framework de modèles de plus en plus détaillé. Cela sera indispensable pour pouvoir monitorer, faire évoluer les différents modèles sans être dépendant d'une ressource humaine particulière et plus globalement minimiser les risques de dérive. La gouvernance des modèles doit donc permettre de rendre le travail des data scientists plus simple, plus automatisé et, en fin de compte, plus productif.
Enfin au delà de la productivité, la gouvernance des modèles permettra de s'assurer que les modèles data sont déployés d'une manière sûre, certifiable et vérifiable. En effet, la tendance sociétale consistant à s'assurer, qu'en plus d'être efficaces, les modèles data sont éthiques, impartiaux et vérifiables devrait continuer à se renforcer. Ce besoin d'une "IA de confiance" constitue une raison supplémentaire de gouverner ses modèles en production.

Pour mener à bien le déploiement de la gouvernance des modèles et pour optimiser leur bonne appropriation et utilisation, il est nécessaire de ne pas oublier l'indispensable conduite du changement à mener. Former les équipe métiers utilisatrices de ces modèles à la compréhension des enjeux et des risques de dérive et au fonctionnement des modèles est essentiel pour qu'elles s'approprient complètement les modèles data et les intègrent dans leur fonctionnement quotidien. De la même manière que la gouvernance des données est d'abord une aventure humaine et un travail de formation, de conviction et d'adhésion de l'organisation à la transformation data, la gouvernance des modèles devra reposer sur un programme de conduite du changement ad hoc. Ce point est clé pour lever les freins humains à l'adoption des modèles en production, garantir leur bonne exploitation par les utilisateurs Métier concernés et les intégrer pleinement dans le cycle de vie des modèles.

QUESTIONS À YOUSSEF BENCHEKROUN (VEEPEE)

Youssef Benchekroun est manager et lead data scientist chez Veepee. Il est en charge d'un service de personnalisation du contenu pour les clients. Veepee travaille également sur des projets de prévision des volumes de vente, pricing des produits et labellisation automatique de contenu.

Quelle est l'organisation data mise en place chez Veepee ?

L'équipe data a été mise en place en 2018 et est composée aujourd'hui de 70 personnes, principalement en France mais également en Espagne et en Belgique. Les profils sont variés, avec une majorité de data scientists, des data engineers, des data analysts et des experts BI. Nos équipes sont orientées produit et travaillent en mode agile, nous gérons ainsi les projets de bout en bout, sans transfert de responsabilité à l'IT une fois les modèles en production.

Avez-vous mis en place une gouvernance des modèles en production ?

Nous avons instauré des indicateurs de performance business des algorithmes qui mesurent par exemple le taux de conversion client ou le CA généré par les différents modèles. Lorsque nous constatons une dérive ou une anomalie, des personnes d'astreinte analysent la situation. Soit ils corrigent le problème de manière agile, soit ils basculent les flux vers un autre modèle le temps de corriger le problème. Si besoin, nous ré-entraînons le modèle et nous le validons avec l'ensemble des parties prenantes avant de le remettre en production.

Pensez-vous que cette organisation soit amenée à évoluer à l'avenir ?

L'organisation actuelle - orientée produit - ne nécessite pas de gouvernance particulière. Demain, avec des équipes plus grandes, plus matures et éclatées sur différents projets, il pourra y avoir un intérêt de mettre en place une gouvernance spécifique.

Quels sont les défis pour gérer des modèles en production ?

La première étape est de s'assurer de la reproductibilité des modèles. Pour cela, nous stockons les paramètres d'entraînement (hyper paramètres, données d'entraînement, etc.) grâce aux outils de model management de Google Cloud Plateform. La deuxième étape est la mise en place d'indicateurs et d'alertes pour être capable de réagir le plus rapidement possible en cas de dérive ou de dysfonctionnement.

Quels sont les risques associés à une dérive ?

Nos modèles de personnalisation du contenu ont un impact direct sur le chiffre d'affaires. Le principal risque est donc interne : manque à gagner pour Veepee. Il y a également un risque d'image si les recommandations ne sont pas pertinentes mais celui-ci est moindre. Nous n'avons pas de risque légaux, juridiques ou réglementaires.

Avez-vous constaté des dérives ?

Nous n'avons pas constaté de dérive pure du modèle à ce jour. La problématique que nous rencontrons parfois provient de la qualité des données qui peut fluctuer. Si la donnée change, cela aura un impact direct sur les modèles. Nous avons donc besoin de robustifier les données en entrée de nos modèles afin de limiter les impacts pour le business.

D'autre part, nous cherchons à développer des algorithmes robustes par design au dérives potentielles (principalement au problème de « cold start » dans le cadre de système de recommandations).

Etes-vous en lien avec l'équipe en charge de la gouvernance de la donnée ?

L'équipe en charge de la gouvernance de la donnée est rattachée à l'équipe data et s'occupe de désiloter les données et d'instaurer des contrats d'interface. Ceci permet de faire le lien entre consommateurs et fournisseurs de données.

Capitalisez-vous sur les modèles existants ?

Étant organisé en équipes agile, la communication se fait naturellement de manière très fluide, ce qui permet de s'échanger facilement les bonnes pratiques en gré à gré. Nous utilisons également GitLab. Par ailleurs, nous essayons de mutualiser les outils au sein des équipes, avec pour cible moyen terme de mettre en place des API des services pour faciliter les échanges de données.

Quels sont vos prochains grands chantiers ?

Notre priorité est la data validation afin de sécuriser la qualité des données en entrée de nos modèles. Nous travaillons avec Tensor Flow Data Validation dont la promesse est de détecter les changements de distribution et les anomalies avant que la donnée soit ingérée. Notre prochain chantier sera de mettre en place des KPIs plus data science que business afin d'avoir un coup d'avance : identifier les dérives avant qu'elles n'impactent le métier.

QUESTIONS À EMMANUEL SOUQUE (BPCE)

Emmanuel Souque est responsable pilotage projets et coordination à la Direction des Risques du groupe BPCE.

Qu'est-ce que le Model Risk Management (MRM) ?

Les modèles sont pour une entreprise un patrimoine intellectuel gage de compétitivité.

Cependant, la multiplicité et l'intégration croissante des modèles dans les processus de décision ou d'évaluation génèrent un risque de modèle. La mise en place d'un cadre MRM vise à considérer ce risque de modèle comme un risque à part entière avec les implications que cela comporte en termes d'identification, d'évaluation et de pilotage tout au long du cycle de vie du modèle. L'inventaire des modèles permet d'avoir une vision globale des modèles et des usages associés afin de les inscrire dans un cadre global, robuste et efficace permettant de s'assurer du niveau de qualité attendu et requis.

Le premier objectif est donc d'identifier ces risques de manière transverse. Le second est d'en avoir la maîtrise, soit en l'acceptant tel quel, soit en prenant des actions visant à le minimiser.

En corollaire, le MRM va permettre d'identifier des investissements à réaliser pour améliorer la qualité des modèle (sur les data, les process, le périmètre, etc.), et s'inscrire dans une logique d'amélioration continue pour faciliter la prise de décision.

Pour quels modèles utilisez-vous ce cadre ?

De fait, nous sommes plus avancés sur les modèles internes de mesure du risque de crédit utilisés pour le calcul des exigences de fonds propres puisque nous avons des obligations réglementaires fortes à respecter à ce niveau là depuis de nombreuses années. Néanmoins, nous avons voulu viser large en termes de familles : modèles de mesure de risque (risques de taux et de liquidité, conformité, etc.), modèles commerciaux, avec des modèles de différentes nature (statistique, IA,..) ceci qui nécessite un cadre modulable et adaptable qui s'appuie également sur des dispositifs existants (BCBS 239 sur les données, risque opérationnel, etc.).

L'arrivée de modèles adaptatifs nécessite par ailleurs une logique de surveillance en continue, ce qui est relativement nouveau et différent par rapport à des modèles qui suivent un cycle de vie plus long avec des back-testings annuels.

Quels sont les risques associés à ces modèles ?

Nous avons toujours suivi les dérives classiques sur des modèles de crédit par exemple où des ré-entraînements ou adaptations de méthodologies peuvent s'avérer nécessaires. Nous sommes également potentiellement sujets à des sanctions financières en cas de d'insuffisances dans les modèles et procédures liés à la conformité (anti-blanchiment, financement du terrorisme…).

L'inventaire MRM des modèles permet de suivre plus finement le cas des "modèles suiveurs" c'est-à-dire une réaction en chaîne lorsqu'un problème apparaît sur un modèle en amont qui alimente ensuite un modèle en aval. On peut également identifier plus rapidement les impacts d'un problème de qualité de données, ou encore de problème d'implémentation.

Quelle organisation avez-vous mis en place ?

Natixis, notre banque de grande clientèle qui a des implantations aux US est soumise à une réglementation stricte sur le sujet, celle-ci dispose donc d'une fonction MRM déjà bien installée. Nous sommes en train de généraliser cette fonction MRM au niveau de l'ensemble du Groupe BPCE, rattachée à la Direction des Risques. Une gestion efficace du risque de modèle est structurée autour de 3 lignes de défense. La première, au sein des métiers, conçoit et utilise le modèle. Elle doit reconnaître et accepter le risque du modèle qu'elle génère. La seconde ligne est là pour valider et contrôler les modèles, en donnant un avis sur le modèle et sur les risques associés. Enfin une troisième ligne s'occupe de réaliser l'audit du cadre de suivi du risque de modèle et peut effectuer des revues indépendantes également. La fonction MRM, au niveau de la 2ème ligne de défense est là pour s'assurer que le processus de validation fonctionne.

Et en termes d'outils ?

Aujourd'hui, nous travaillons principalement via Excel, avec la volonté de s'outiller prochainement, plutôt avec une solution orientée processus

adaptée à nos besoins. Ces outils sont relativement simples : base de données pour gérer les inventaires, outil de planification des revues, gestion des processus...

Quelles sont les difficultés rencontrées dans la mises en place du MRM et dans sa gestion au quotidien ?

Il s'agit principalement de difficultés humaines et organisationnelles plutôt que technologiques. Il y a un vrai effort de pédagogie à faire afin d'expliquer l'intérêt de cette approche aux premières lignes pour ne pas qu'elle soit considérée comme une contrainte de plus mais bien comme un accompagnement permettant d'avoir un autre regard sur les risques associés à notre activité. Certaines équipes sont naturellement plus sensibilisées à ces risques. La co-construction des normes et des standards avec eux est une bonne manière de les impliquer.

Quelles sont selon vous les bonnes pratiques à mettre en place ?

La première étape est de lister les modèles : comprendre leurs usages, leur utilisation, les responsabilités, etc. Cela est nécessaire pour dimensionner un framework MRM. La seconde est de réutiliser ce qui existe déjà en matière de risques de modèle et de l'adapter à son contexte : pas besoin de réinventer la roue, la norme BCBS 239 traite déjà des aspects sur les données, la fiabilité des processus est déjà suivie par le risque opérationnel, etc. Enfin, il est nécessaire de prioriser les modèles, par exemple en fonction de leur matérialité - importance relative - (contexte, volumes en jeu, impacts financiers) et de leur santé (précision, environnement d'utilisation, qualité de la donnée), puis de les gérer en fonction de cette priorisation en mettant en place des fréquences de revue adaptées.

Peut-on imaginer des algorithmes qui s'occuperont du MRM ?

Pour certaines parties oui. Le Robot Process Automation (RPA) pourrait par exemple être utilisé pour automatiser certaines parties du processus : revue, test de robustesse, contrôle de données, etc. En revanche, les phases d'inventaire et de suivi qualitatif, ainsi que les aspects organisationnels et humains - qui représentent la vraie valeur ajoutée - seront toujours effectués par des humains.

POINTS CLÉS ET PERSPECTIVES

Ces différents chapitres apportent des éléments de réponses sur la gestion du cycle de vie d'un modèle. Les outils et les méthodes présentées, si elles sont convenablement appliquées et bien sûr adaptées à chaque contexte, permettront d'instaurer un climat de confiance dans le monde de l'intelligence artificielle et d'en tirer le ROI espéré.

On peut retenir quelques points principaux :

- Anticiper dès le démarrage projet la mise en production, et développer un pipeline complet avec des modèles simples, de façon à privilégier l'industrialisation itérative au lieu d'expérimenter sans limites. Dès ce pipeline simple mis en place, mettre en place les KPI de monitoring de modèles ainsi que la politique de réentrainement, calcul du ROI.
- Mettre en place un solide monitoring de la dérive des données, qui est prévisible et dont les raisons doivent être diagnostiquées.
- Tous les modèles dérivent. L'IA est certe une innovation de rupture, cela ne l'empêche pas d'être soumise à la même rigueur scientifique que les méthodes de modélisation traditionnelles.
- La multiplication de modèles automatiques et dépendants des données de l'entreprise pose un nouveau risque opérationnel qu'il faudra quantifier et maîtriser. Le Model Risk Management est une base de réflexion très intéressant pour la mise en place d'une gouvernance des modèles.

Avec la multiplication des modèles IA en production, la problématique du cycle de vie sera de plus en plus prégnante, et en réalité un pré-requis de succès des projets, à l'heure actuelle encore sous-estimé. Les interactions entre équipes Data Scientists, DSI, et utilisateurs des modèles, seront à définir dans un RACI propre à chaque organisation, mais où certaines bonnes pratiques sont tout de même déjà identifiables.

RÉFÉRENCES COMPLÉMENTAIRES

CHAPITRE 2

1. http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions
2. https://docs.microsoft.com/en-us/azure/aks/
3. https://cloud.google.com/ai-platform/
4. https://cloud.google.com/stackdriver/
5. https://prometheus.io/docs/introduction/overview/
6. https://hydrosphere.io/sonar/
7. https://logz.io/
8. https://www.splunk.com/
9. https://prodi.gy/
10. https://supervise.ly/

CHAPITRE 4

11. https://www.jasonshulmanstudio.com/photographs-of-films
12. https://dl.acm.org/citation.cfm?id=2523813
13. https://ieeexplore.ieee.org/document/1647649
14. https://ieeexplore.ieee.org/document/5975223
15. https://www.sciencedirect.com/science/article/pii/S0378375813000633
16. https://dcor.readthedocs.io/en/latest/index.html
17. https://github.com/AxeldeRomblay/MLBox
18. https://link.springer.com/chapter/10.1007/11893318_7
19. https://www.researchgate.net/publication/300125762_An_Overview_of_Concept_Drift_Applications
20. https://arxiv.org/abs/1010.4784
21. https://dl.acm.org/citation.cfm?id=347107
22. https://scikit-multiflow.github.io/
23. https://www.sciencedirect.com/science/article/abs/pii/S1566253516302329

CHAPITRE 5

6. https://towardsdatascience.com/a-concrete-application-of-topological-data-analysis-86b89aa27586
7. https://learning-analytics.info/journals/index.php/JLA/article/view/5196/6089
8. http://burrsettles.com/pub/settles.activelearning.pdf
9. https://medium.com/nanonets/how-to-use-deep-learning-when-you-have-limited-data-part-2-data-augmentation-c26971dc8ced
10. http://mathematiques.univ-lille1.fr/digitalAssets/29/29427_71-3.pdf
11. https://towardsdatascience.com/understanding-generative-adversarial-networks-gans-cd6e4651a29
12. https://arxiv.org/pdf/1803.00401.pdf
13. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5355231/

Après de nombreuses réflexions concernant l'industrialisation et l'intelligibilité des modèles de Machine et Deep Learning, les projets data font aujourd'hui face à la problématique du maintien en conditions opérationnelles. C'est un sujet d'autant plus complexe qu'il implique la coordination entre plusieurs métiers (data scientists, data engineers, DSI, utilisateurs finaux) afin d'assurer des modèles robustes, un ROI effectif et une réactivité suffisante en cas de problème.

La recherche autour des problématiques de sensibilité et de robustesse des modèles est en pleine expansion, de même que les outils de versionning et monitoring. Chez Quantmetry, nous sommes convaincus qu'il est crucial pour les entreprises de mettre en place des standards à la fois techniques, méthodologiques et organisationnels pour faire face à cet enjeu. Ainsi, nous avons créé un pôle d'expertise R&D afin d'approfondir ce sujet et vous proposer des éléments de réponse.

Ce livre blanc aborde ce sujet sous ces trois axes, proposant des éléments de réponse théoriques et concrets, complétés de retours d'expérience variés d'entreprises et de chercheurs.

Bonne lecture !