

Validation et gestion des données

MGL 7811: Ingénierie logicielle des systèmes d'intelligence artificielle



G I G O

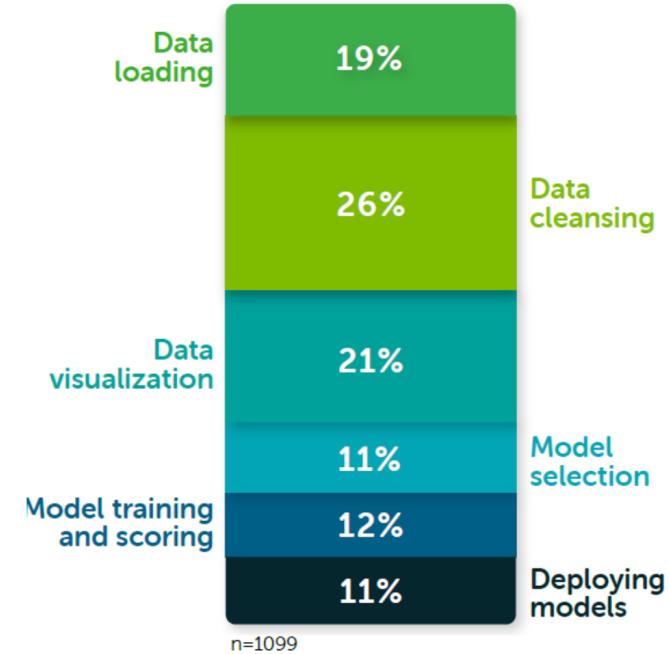


Garbage In Garbage Out



La qualité des données est primordiale pour la qualité ML

- Un modèle d'apprentissage automatique n'est aussi bon que les données qu'il est alimenté
- Les scientifiques des données ont passé beaucoup plus de temps à assurer **la qualité des données** qu'à travailler avec les modèles



How data scientists spend their time (Image courtesy Anaconda “[2020 State of Data Science: Moving From Hype Toward Maturity](#).”)

Les mauvaises données sont pires que l'absence de données

- Zillow
 - Prédire les prix des maisons
 - Achetez des maisons à un prix inférieur aux prévisions
 - Vendre au prix prévu
 - Profit
- La précision du modèle a commencé à se dégrader
 - Problèmes de qualité des données
 - Augmentation de l'erreur de prédiction
 - **Perte de 300 millions de dollars**

Sharan Kumar Ravindran
Nov 5, 2021 · 9 min read · Member-only · Listen

Invaluable Data Science Lessons To Learn From The Failure of Zillow's Flipping Business

What went wrong?

<https://towardsdatascience.com/invaluable-data-science-lessons-to-learn-from-the-failure-of-zillows-flipping-business-25fdc218a62>

BUSINESS

Zillow will stop buying and renovating homes and cut 25% of its workforce

November 3, 2021 · 1:44 PM ET

<https://www.npr.org/2021/11/03/1051941654/zillow-will-stop-buying-and-renovating-homes-and-cut-25-of-its-workforce>

Les mauvaises données sont pires que l'absence de données (cont.)

- Amazon système d'embauche
 - Les entreprises reçoivent des tonnes de CV
 - Utilisez ML pour classer les CV
 - Choisissez les meilleurs CV pour l'entrevue
 - Profit

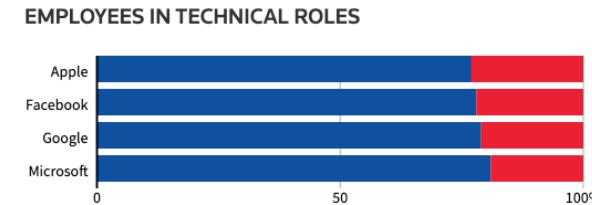
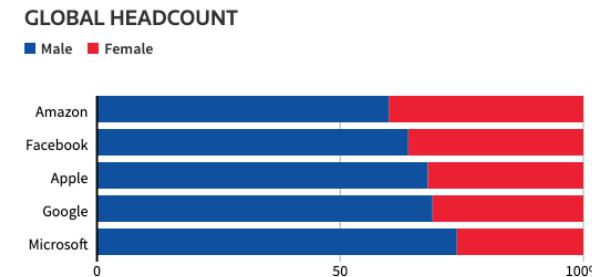
WORLD

Amazon ditches AI recruiting tool that didn't like women

By Jeffrey Dastin • Reuters
Posted October 10, 2018 6:46 am

<https://globalnews.ca/news/4532172/amazon-jobs-ai-bias/>

Qualité des données



Note: Amazon does not disclose the gender breakdown of its technical workforce.
Source: Latest data available from the companies, since 2017.
By Han Huang | REUTERS GRAPHICS

Les mauvaises données sont pires que l'absence de données (cont.)

- Amazon Rekognition scan
 - Entrez une image
 - Faire correspondre l'image aux bases de données mugshot
 - Profit (?)

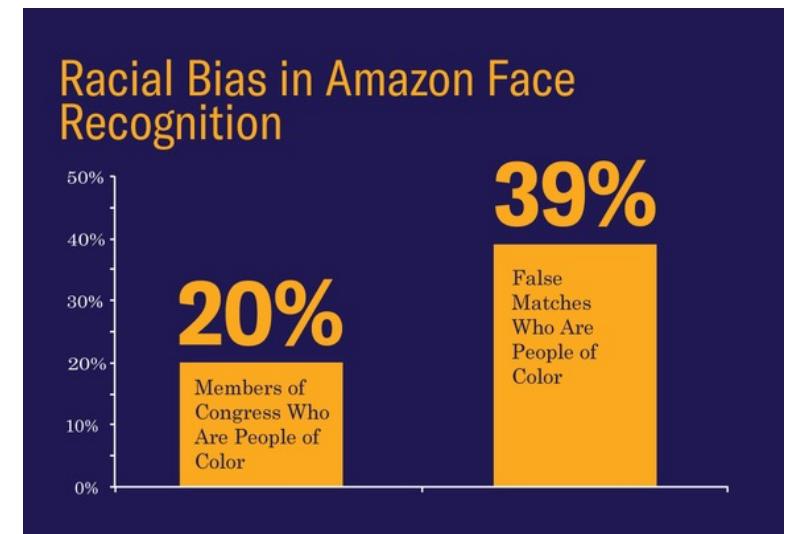
NEWS & COMMENTARY

Amazon's Face Recognition Falsely Matched 28 Members of Congress With Mugshots



<https://www.aclu.org/news/privacy-technology/amazons-face-recognition-falsely-matched-28>

Qualité des données



Plusieurs autres exemples

The screenshots show the genderify AI-based gender verification interface. Both displays the same text: "Our real-time AI-based gender verification from name, username or email. Get the best of our unique solution that's the only one of its kind available in the market. Go ahead, **try below!**".

Screenshot 1 (Left): Input field contains "Ariel", button is "Check". Below: Male: 49.00% Female: 51.00%. A link to suggest results and a "Submit" button.

Gender	Percentage
Male	49.00%
Female	51.00%

Screenshot 2 (Right): Input field contains "Dr. Ariel", button is "Check". Below: Male: 93.20% Female: 6.80%. A link to suggest results and a "Submit" button.

Gender	Percentage
Male	93.20%
Female	6.80%

<https://twitter.com/fisadev/status/1288327018522779648/photo/2>

Top 10 ML Model Failures You Should Know About

Deepchecks Community Blog | October, 18 2022

<https://deepchecks.com/top-10-ml-model-failures-you-should-know-about/>

Data quality and AI safety: 4 ways bad data affects AI and how to avoid it

Jane Hillman | 30 January 2023

<https://www.prolific.co/blog/data-quality-and-ai-safety>

Les mauvaises données ne génèrent pas d'erreurs

- Les erreurs logiciels sont identifier pour:
 - Stack traces
 - Error logs
 - etc.

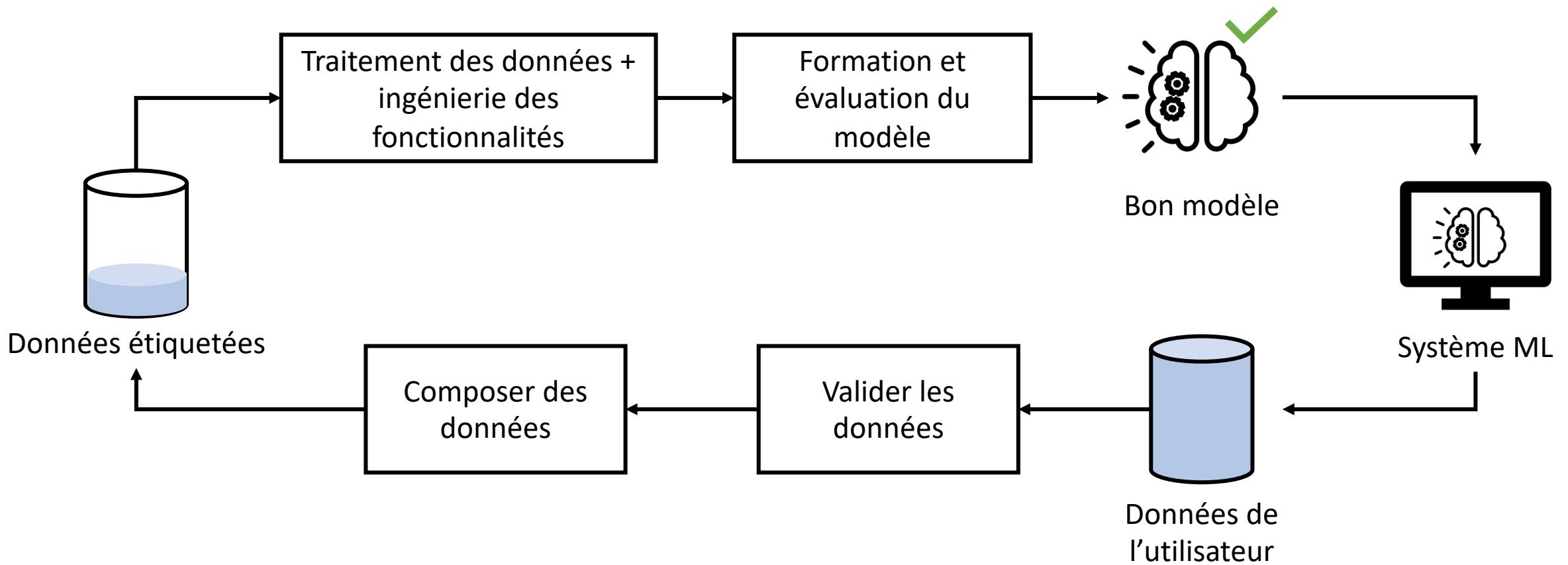
Comment identifier les mauvaises données?

- Le meilleur scénario:
 - Erreurs dans le pipeline de traitement
- Scénario typique:
 - ???

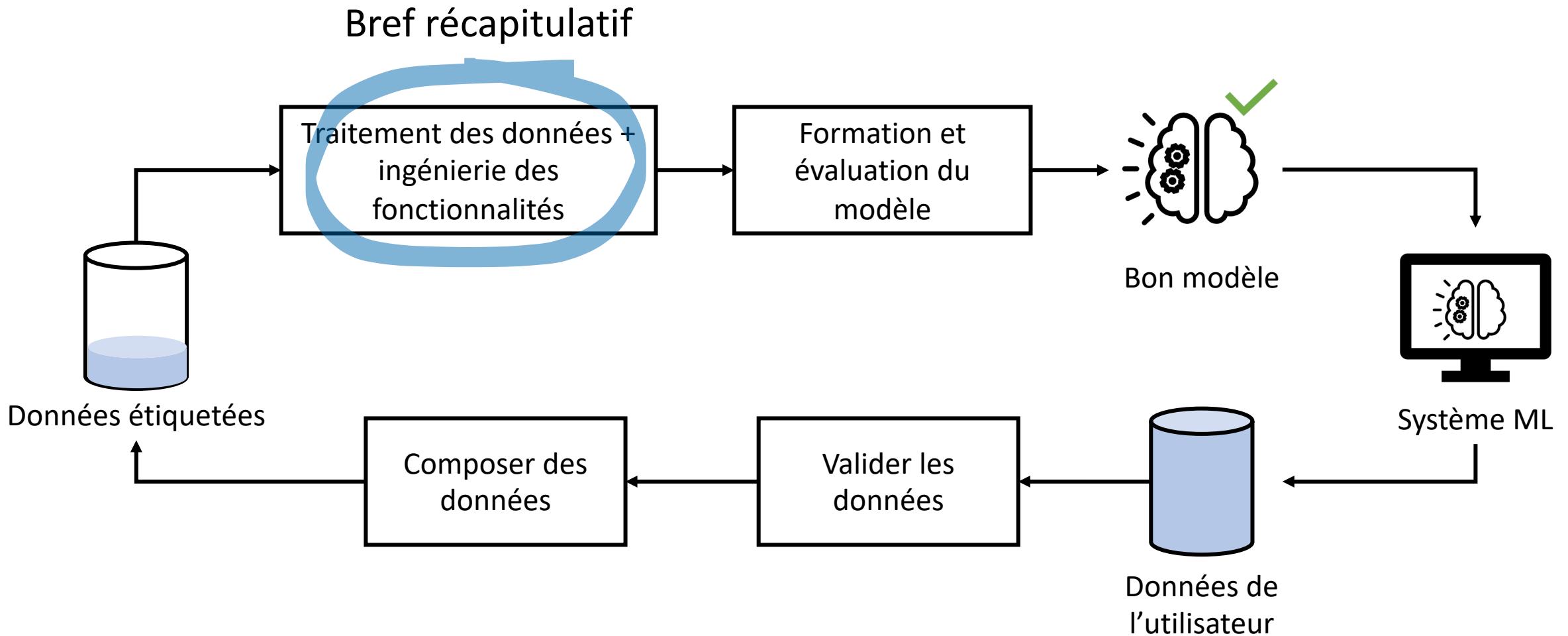
Notre objectif



Pipeline de données des systèmes ML



Pipeline de données des systèmes ML



Traitement**t des données (un bref récapitulatif)**

Exploration des données

Identifier les problèmes dans les données:

- Valeurs manquantes
- Distribution asymétrique (possibilité de biais)
- Valeurs non documentées
- Mauvaise couverture des données

Identifier le type de problème:

- Distribution des variables cibles
 - E.g., Problème de classification équilibré ou déséquilibré?

Traitemen~~t~~ des données

Problème: Les données brutes sont bruyantes

- Plus difficile à modéliser, à prédire et à expliquer

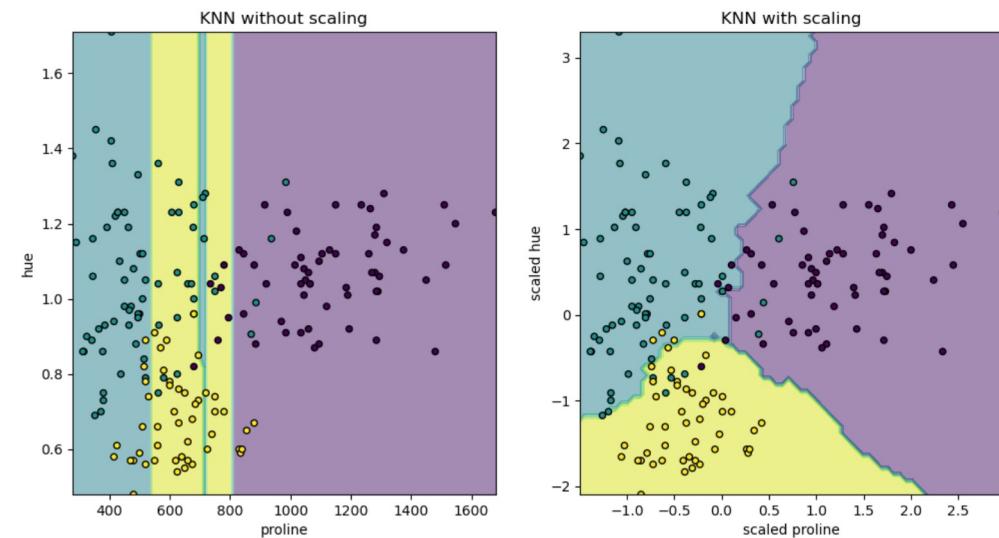
Des solutions:

- Gérer les valeurs manquantes
- Anonymiser les données (données sensibles à la confidentialité)
- Supprimer les valeurs aberrantes
- Spécifier la mise en forme des données

Transformation des données

Problème: Les fonctionnalités sont présentes dans différentes gammes et distributions

- Affecter la performance des modèles
- Plus difficile à expliquer
- Des solutions
 - Feature scaling
 - Feature normalization
 - Dimensionality reduction



Ingénierie des fonctionnalités

- **Problème:** Les fonctionnalités brutes ne sont peut-être pas idéales pour notre problème de domaine
- Des solutions: Ajouter les connaissances du domaine
 - Regroupez les données
 - E.g., Numerical age into age groups (teenagers, young adult, ...)
 - Feature encoding
 - Coder des fonctionnalités catégorielles (one-hot encoding)
 - Créer de nouvelles fonctionnalités
 - D'après des données supplémentaires

Sélection des fonctionnalités

- **Problème:** Toutes les fonctionnalités ne sont pas pertinentes pour notre modèle
 - Augmenter la difficulté de maintenir les données
- Des solutions: Supprimer les fonctionnalités excessives ([Scikit learn methods](#))
 - Supprimer les entités invariantes
 - Supprimer les entités corrélées
 - Si deux entités sont fortement corrélées, supprimez l'une d'entre elles
 - Sélectionnez les meilleures fonctionnalités
 - Utiliser le modèle + la formation et la validation

Division des données (Data Split)

Fractionner les données

- Ensemble de formation et de validation
 - Former et évaluer l'aptitude du modèle
- Ensemble de tests
 - Évaluer le rendement des modèles entraînés

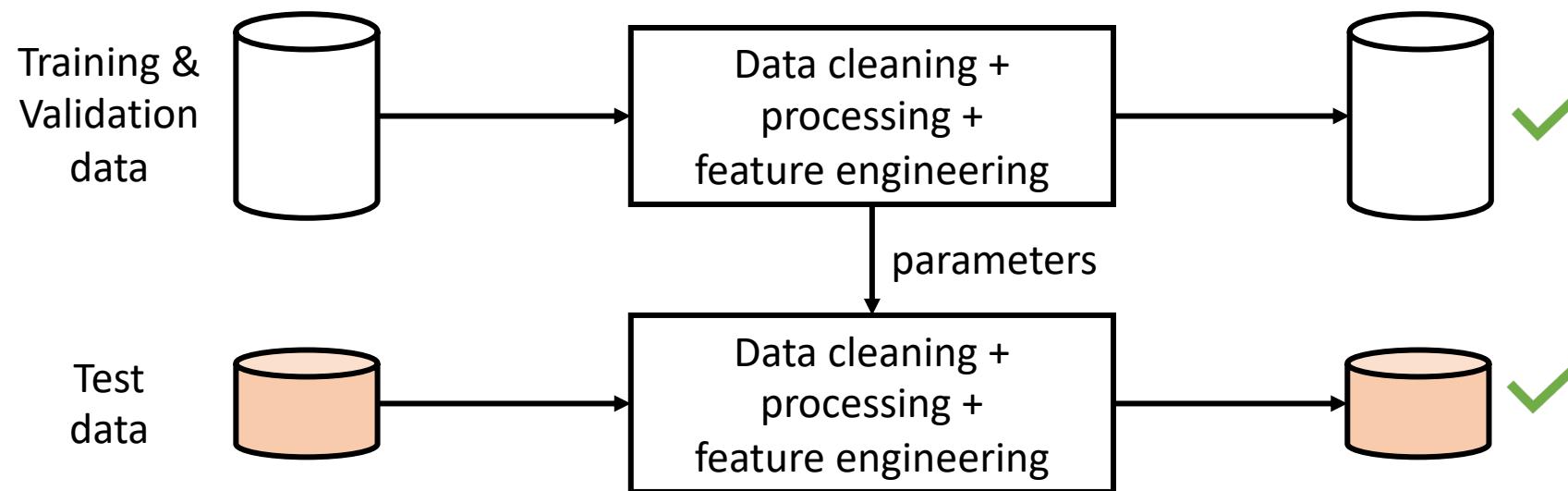
Décision

- Plus de données d'entraînement: mieux pour la performance de la formation
- Plus de données de test: une évaluation plus approfondie

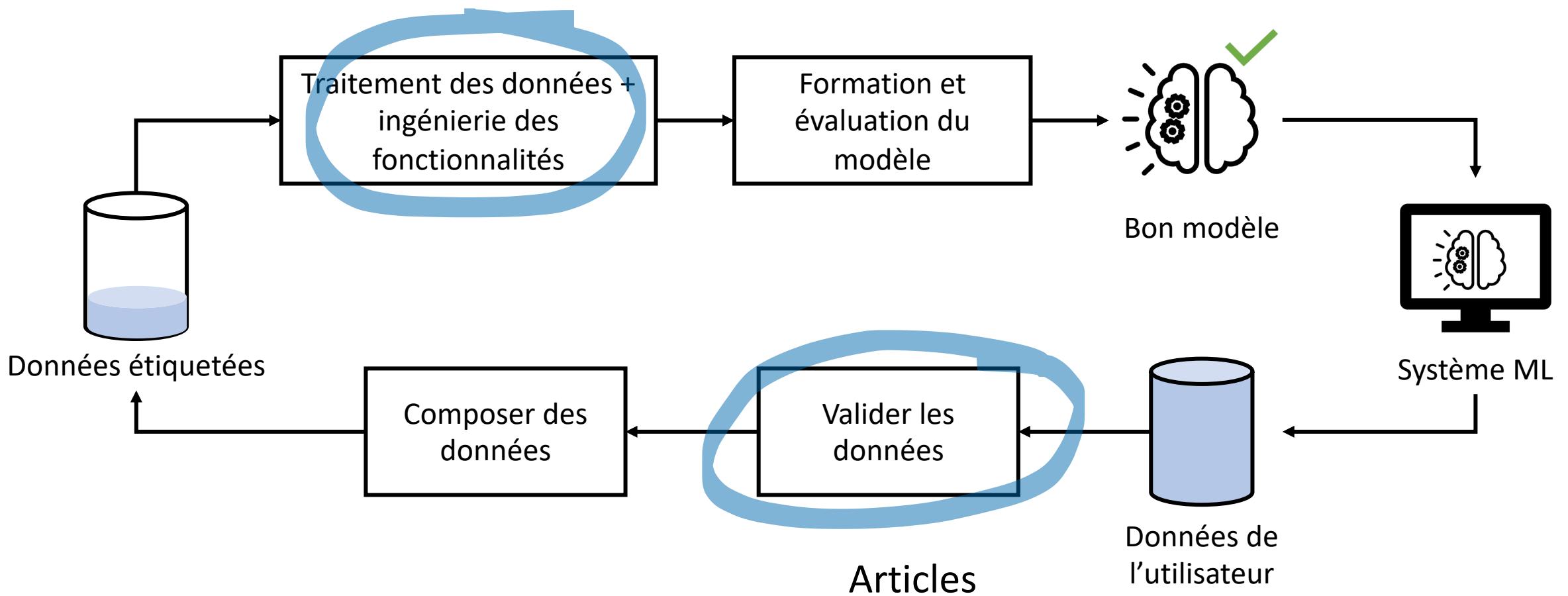
Attention à la division des données

Les données de test **ne doivent jamais** influencer les données d'entraînement

- **Rule of thumb:** diviser les données avant le traitement des données
- Utiliser les paramètres des données d'entraînement pour éclairer la transformation des données de test



Pipeline de données des systèmes ML



Data Linter

- Propose un outil pour **inspecter** les jeux de données afin **d'identifier** les problèmes et **d'améliorer** l'ingénierie de fonctionnalités.

The Data Linter: Lightweight, Automated Sanity Checking for ML Data Sets

Nick Hynes*
Berkeley AI Research (BAIR) Lab
nhynes@berkeley.edu

D. Sculley
Google Brain
dsculley@google.com

Michael Terry
Google Brain
michaelterry@google.com

Abstract

Data cleaning and feature engineering are both common practices when developing machine learning (ML) models. However, developers are not always aware of best practices for preparing or transforming data for a given model type, which can lead to suboptimal representations of input features. To address this issue, we introduce the *data linter*, a new class of ML tool that automatically inspects ML data sets to 1) identify potential issues in the data and 2) suggest potentially useful feature transforms, for a given model type. As with traditional code linting, data linting automatically identifies potential issues or inefficiencies; codifies best practices and educates end-users about these practices through tool use; and can lead to quality improvements. In this paper, we provide a detailed description of data linting, describe our initial implementation of a data linter for deep neural networks, and report results suggesting the utility of using a data linter during ML model design.

Qu'est-ce qu'un Lint?

- **Lint** est un défaut potentiel dans l'expression d'un programme.
- Il existe une multitude de linters dans les logiciels traditionnels
 - erreurs de syntaxe
 - mauvaises pratiques
 - (...)



La motivation

- Les problèmes de données peuvent être identifiés avec une **automatisation légère**:
 - Attributs numériques sur des échelles très différentes
 - Valeurs pour une signification particulière (e.g., -9999 for missing data)
 - Valeurs mal formées
 - ...
- Les auteurs proposent un Data Linter qui implémente tous ces analyses

Study Design

- Tool paper
 - Le concept d'outil
 - L'architecture
 - Évaluation préliminaire
 - Kaggle
 - Google
- Publié sur NIPS

Contribution

- Data Linter
 - Analyse les données d'entraînement de l'utilisateur
 - Statistiques sommaires
 - Exemples individuels
 - Noms des attributs
 - Suggérer des transformations des attributs pour de meilleures performances
 - Disponible à <https://github.com/brain-research/data-linter>
- Architecture et mise en œuvre (haut niveau)
- Évaluation empirique

Architecture

- LintDetectors:
 - Extensible pour l'utilisateur
 - Logique pour détecter le problème
- DataLinter est le moteur qui applique LintDetectors
- LintExplorer
 - Présenter les résultats de manière conviviale
 - Résume les données pour LintDetectors
- Implémenté en tant que bibliothèque Python

Lints de données

- Miscoding (Codage erroné)
 - Transformer les fonctionnalités pour améliorer la performance du modèle
 - Les modèles moins robustes (modèles linéaires) peuvent souffrir de ces problèmes
- Examples (anglais)
 - Number as string -> convert to number
 - Enum as real -> one-hot encoding
 - Long tokenizable strings -> tokenize
 - Circular domain (e.g., day of the week) -> binning
 - Date time as string -> convert to date format
 - ...

Lints de données (cont.)

- Lints: Identifier les valeurs aberrantes
- Attributs non normalisées
 - Large gamme de valeurs, envisager de les normaliser
- “Tailed” distribution
 - Présence de valeurs extremes
- DéTECTeur de signes inhabituels
 - Valeurs négatives, etc.

Lints de données (cont.)

- Linters de données d'erreur d'empaquetage
 - Problèmes d'organisation des données
- Exemples
 - Valeurs en double
 - Exemples vides

Évaluation de l'utilisateur final

- Évaluation de l'utilisateur final
 - Huit ingénieurs logiciels ont utilisé Data Linter
 - Des ingénieurs novices aux ingénieurs expérimentés
- Meilleurs résultats
 - La suggestion de Linter (normalisation des attributs) a conduit à une amélioration de la précision de 0,48 à 0,59.
- Rendement raisonnable
- Feedback: Possibilité de faire taire certains linters

Évaluation de l'ensemble de données Kaggle

- Évaluation sur 600 ensembles de données accessibles au public
- Utilisé l'inférence de type automatique de la bibliothèque pandas

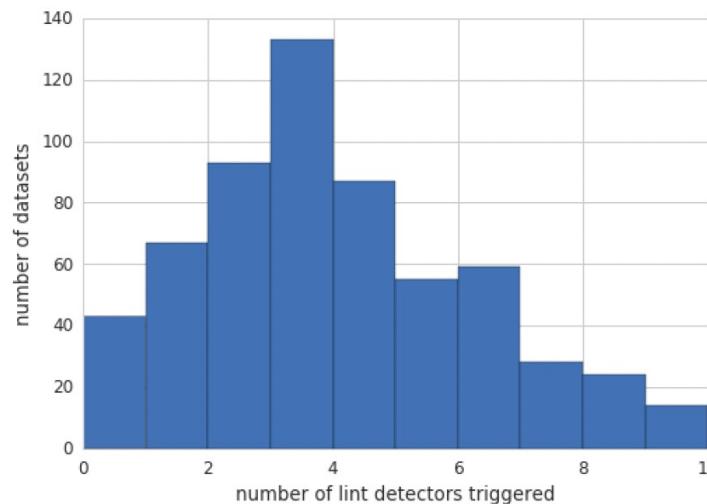


Figure 1: **Histogram of Number of Data Lints Per Data Set, Across 600 Kaggle Data Sets.** Interestingly, only about 7% of the data sets we examined as part of this study had zero data lints and were thus “lint free.” The other 93% had at least one data lint, suggesting the utility of automated checking.

Évaluation de l'ensemble de données Kaggle

- Évaluation sur 600 ensembles de données accessibles au public
- Utilisé l'inférence de type automatique de la bibliothèque pandas

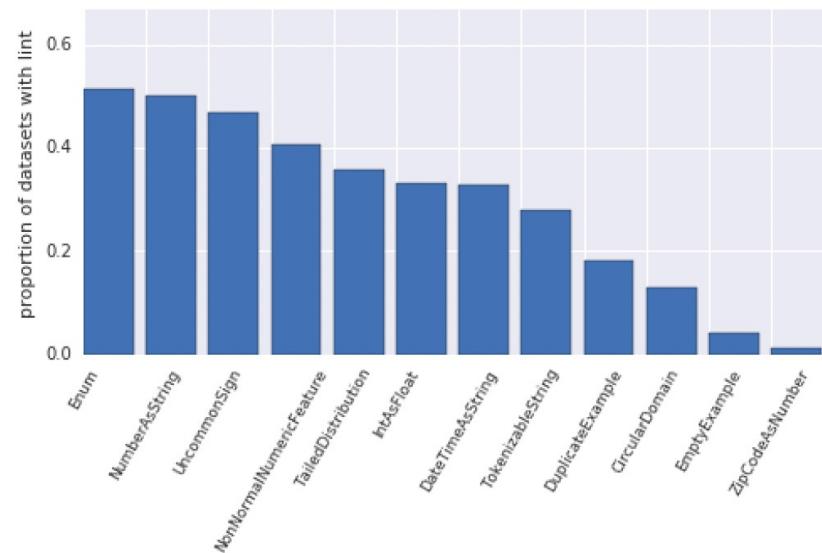


Figure 2: **Frequency of Data Lints Across 600 Kaggle Data Sets.** Some data issues are relatively common, including encoding numeric values as strings and issues around encoding enumerated values. But even rarer issues such as empty examples or potentially problematic encoding of postal codes do occur in the real-world data sets.

Valider les résultats

- Examiner manuellement 35 ensembles de données
 - N'a identifié aucun faux négatif
- La validité de cette analyse est limitée par les connaissances de l'auteur
 - Menaces à la validité?

Discussion ouverte



Data Validation for Machine Learning

- Présente un système de validation des données pour détecter les **anomalies** dans les pipelines ML

DATA VALIDATION FOR MACHINE LEARNING

Eric Breck¹ Neoklis Polyzotis¹ Sudip Roy¹ Steven Euijong Whang² Martin Zinkevich¹

ABSTRACT

Machine learning is a powerful tool for gleaning knowledge from massive amounts of data. While a great deal of machine learning research has focused on improving the accuracy and efficiency of training and inference algorithms, there is less attention in the equally important problem of monitoring the quality of data fed to machine learning. The importance of this problem is hard to dispute: errors in the input data can nullify any benefits on speed and accuracy for training and inference. This argument points to a data-centric approach to machine learning that treats training and serving data as an important production asset, on par with the algorithm and infrastructure used for learning.

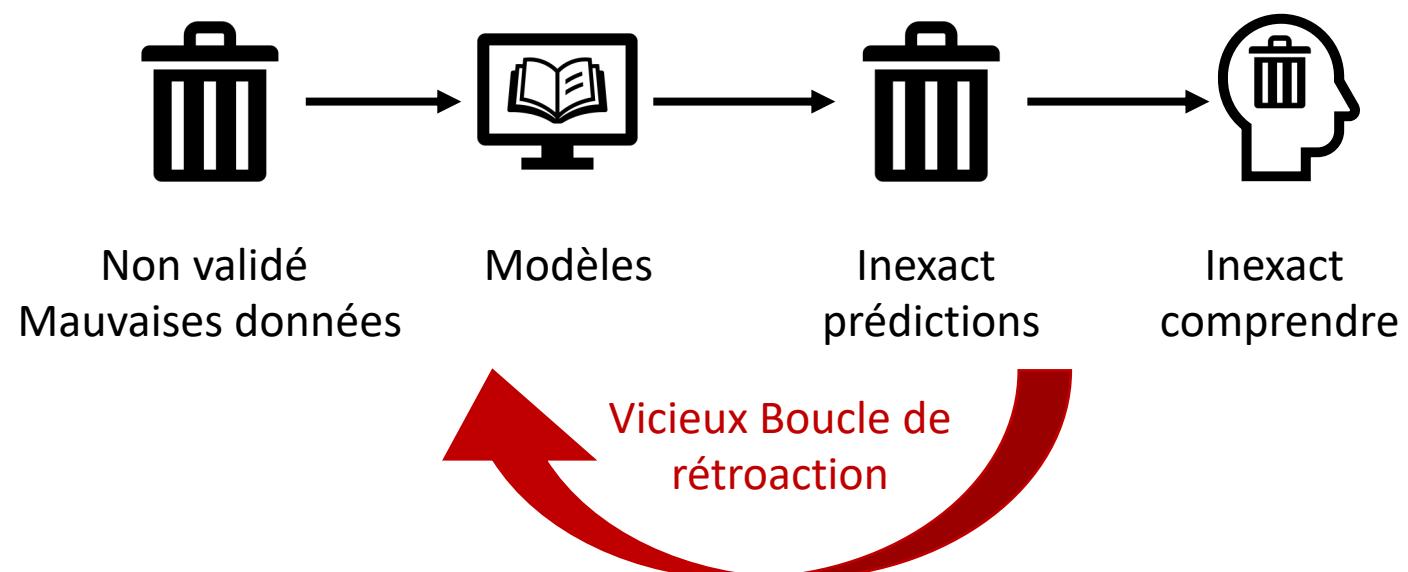
In this paper, we tackle this problem and present a data validation system that is designed to detect anomalies specifically in data fed into machine learning pipelines. This system is deployed in production as an integral part of TFX(Baylor et al., 2017) – an end-to-end machine learning platform at Google. It is used by hundreds of product teams use it to continuously monitor and validate several petabytes of production data per day. We faced several challenges in developing our system, most notably around the ability of ML pipelines to soldier on in the face of unexpected patterns, schema-free data, or training/serving skew. We discuss these challenges, the techniques we used to address them, and the various design choices that we made in implementing the system. Finally, we present evidence from the system’s deployment in production that illustrate the tangible benefits of data validation in the context of ML: early detection of errors, model-quality wins from using better data, savings in engineering hours to debug problems, and a shift towards data-centric workflows in model development.

Motivation

- L'apprentissage automatique est un outil puissant pour glaner des **connaissances** à partir de quantités massives de données.
- Les chercheurs se sont concentrés sur:
 - Efficacité de la formation des modèles
- Il est nécessaire de proposer de meilleures approches pour **surveiller** la qualité des données.

La portée de l'article

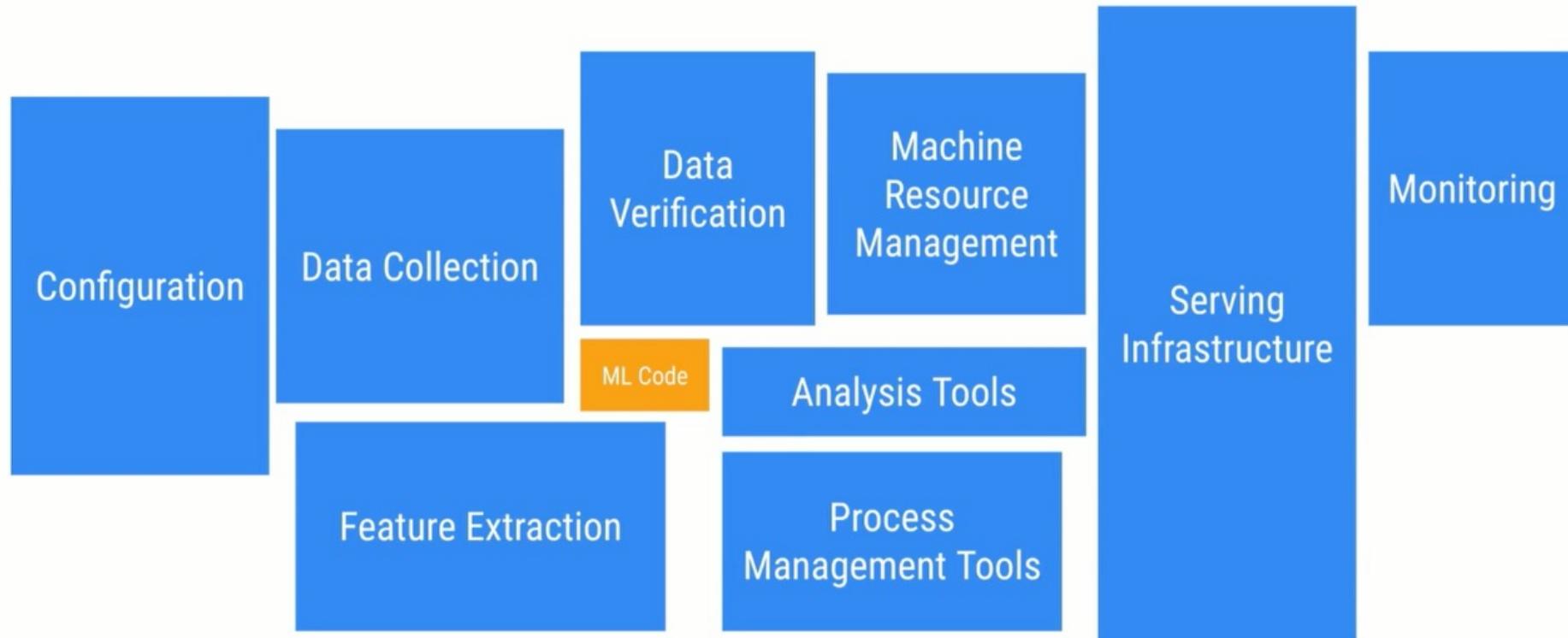
- Se concentre sur le problème de la **validation** des données d'entrée des pipelines ML
- Pourquoi??



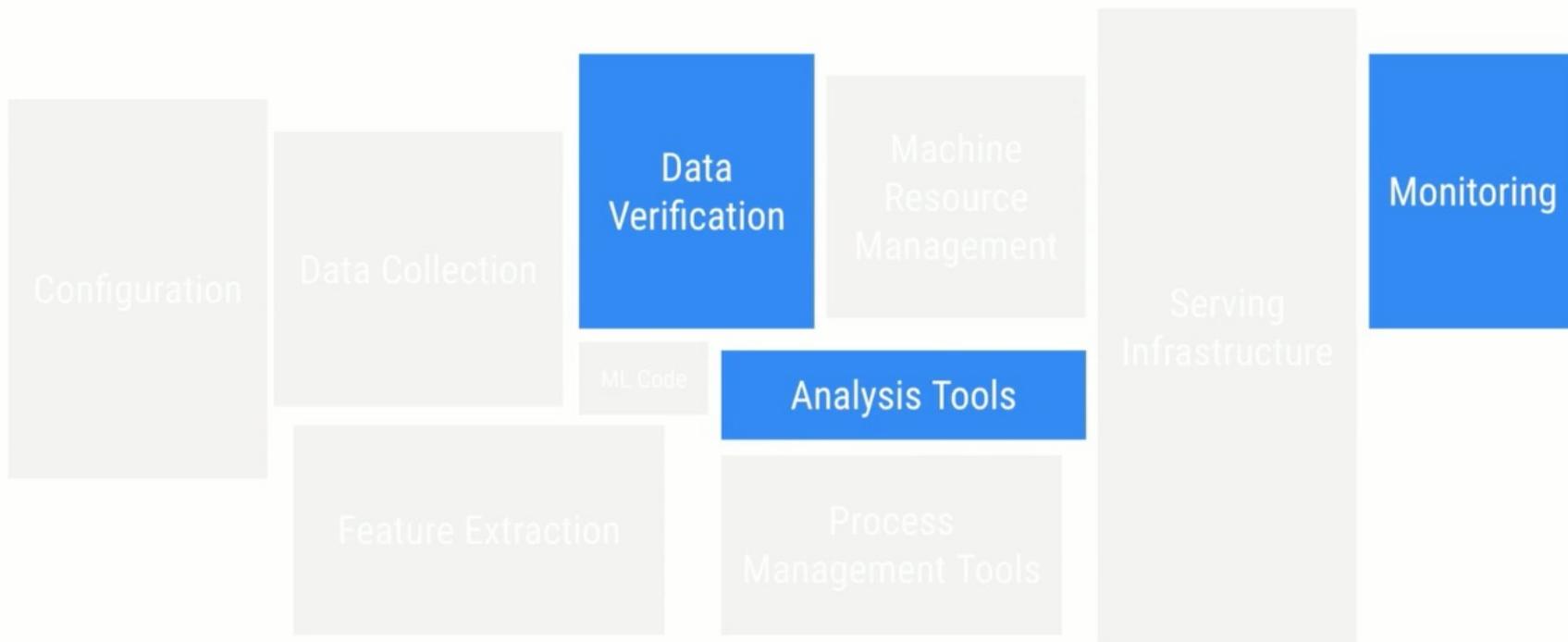
La portée de l'article

- Se concentre sur le problème de la **validation** des données d'entrée des pipelines ML
- Pourquoi?
 - Les petites erreurs dans les données sont **amplifiées!**
- Que faire?
 - Tenter de détecter les erreurs de données au début du processus ML

La portée de l'article

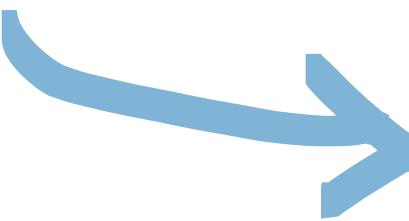


La portée de l'article



Contribution

- Un système de validation des données
 - Bien testé en environnement industriel
 - Déployé dans le cadre de TFX



TensorFlow Extended

- Les bibliothèques sont accessibles au public
 - <https://github.com/tensorflow/data-validation>

Study Design

- “System’s paper”
- Rapport industriel
- Mettre l’accent sur la présentation de l’approche
- Évaluation empirique
 - Déploiement en production
 - Études de cas
- Publié à la conférence SysML

Terminologie du ML

- Training data
 - Serving data
 - Model inference
-
- Training data generation code
 - Serving data generation code

Vue d'ensemble du système

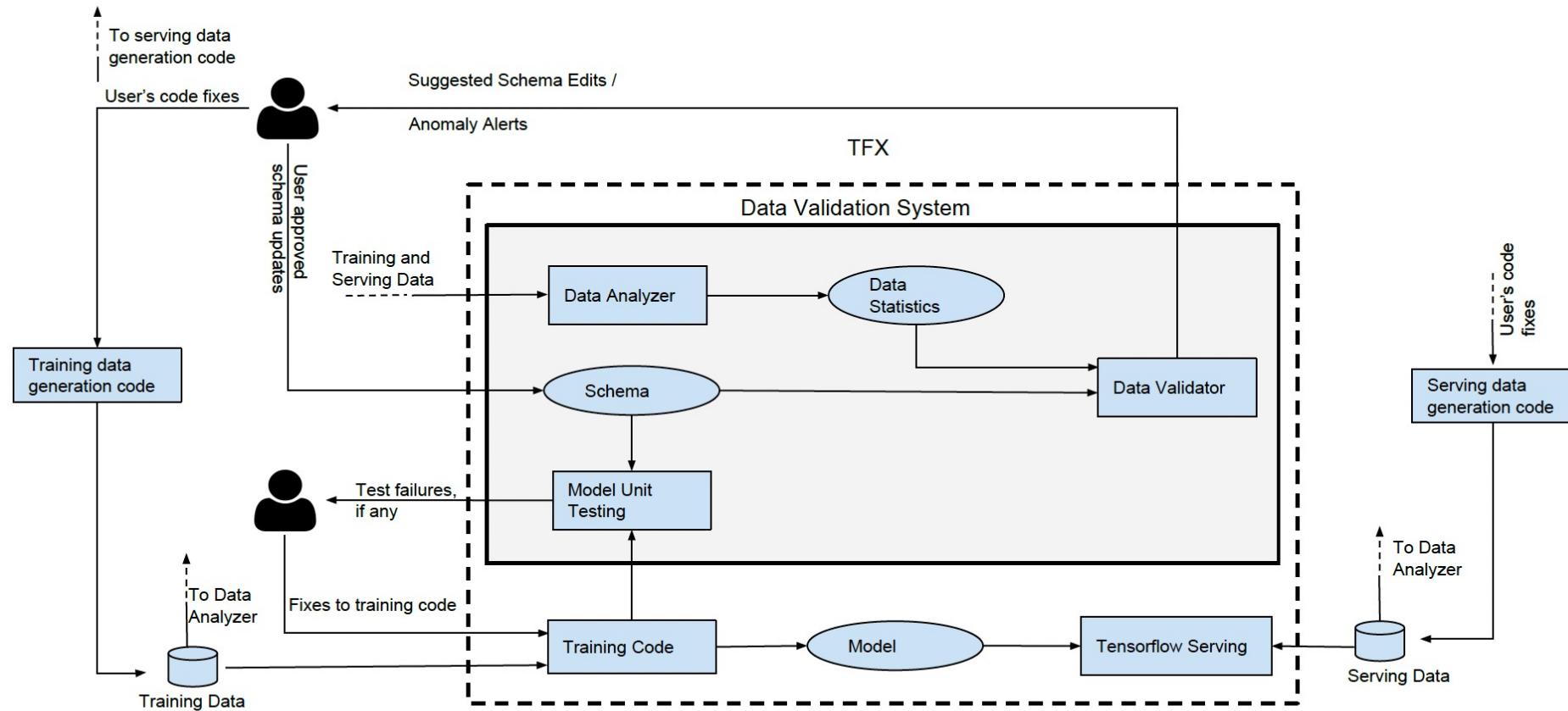


Figure 1: An overview of our data validation system and its integration with TFX.

Vue d'ensemble du système

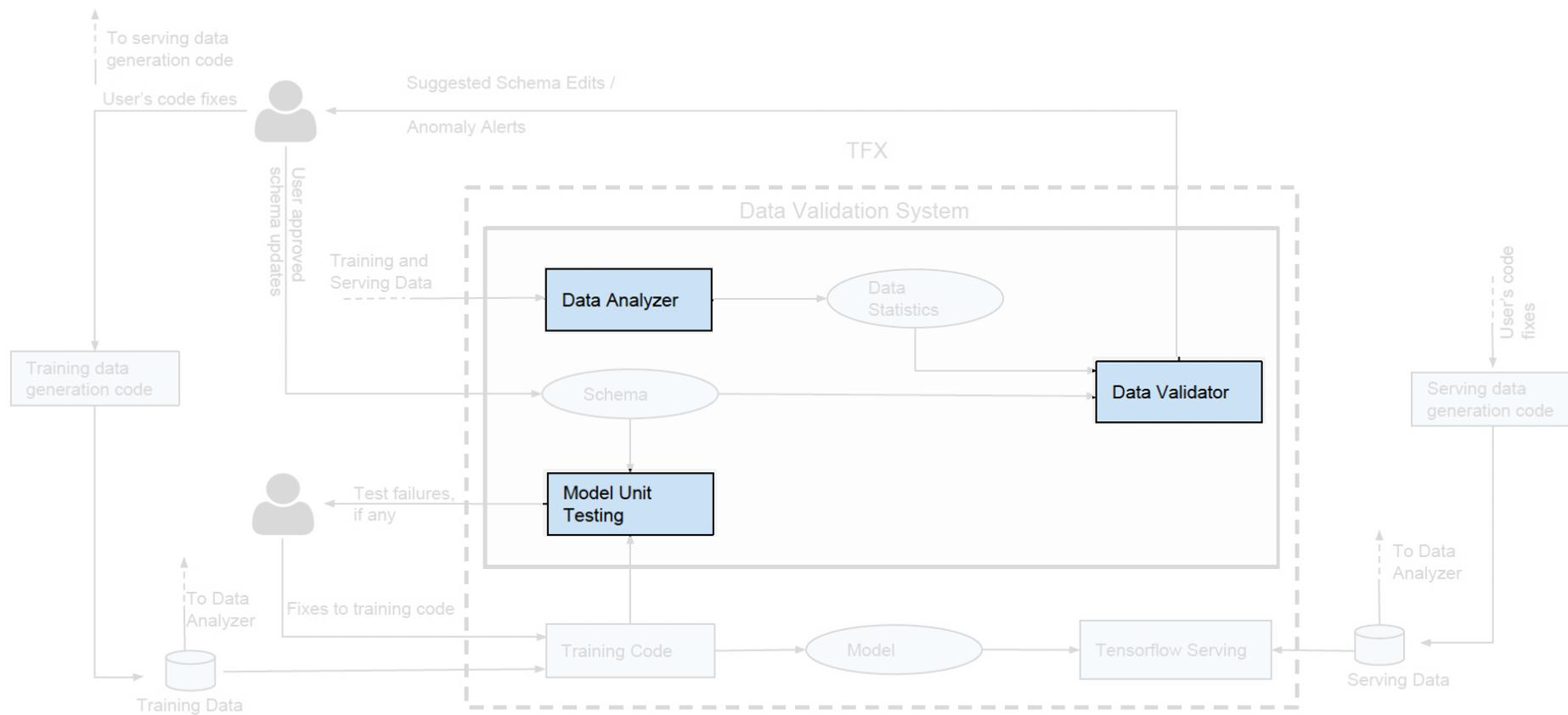
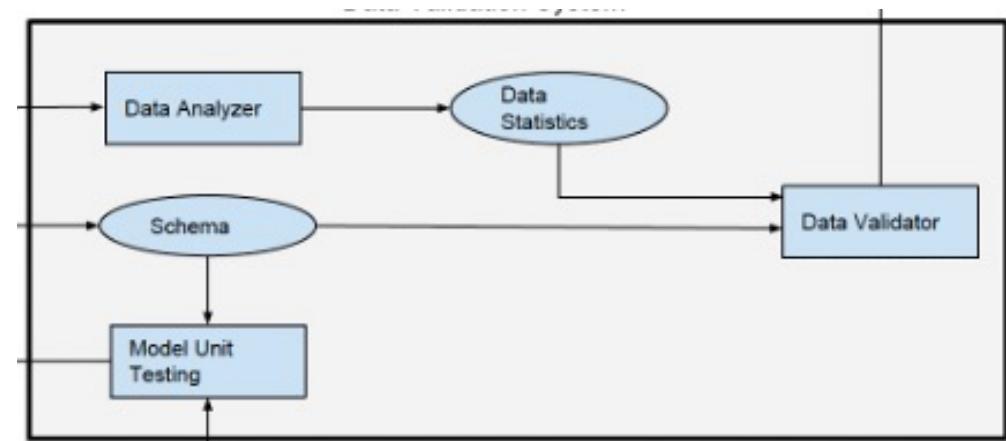


Figure 1: An overview of our data validation system and its integration with TFX.

Composantes

- Analyseur de données
 - Calcul des statistiques de données pour la validation des données
- Validateur de données
 - Vérifie les propriétés des données
- Testeur d'unité de modèle
 - Vérifie les erreurs dans le code d'entraînement à l'aide de données synthétiques



Types de validation des données

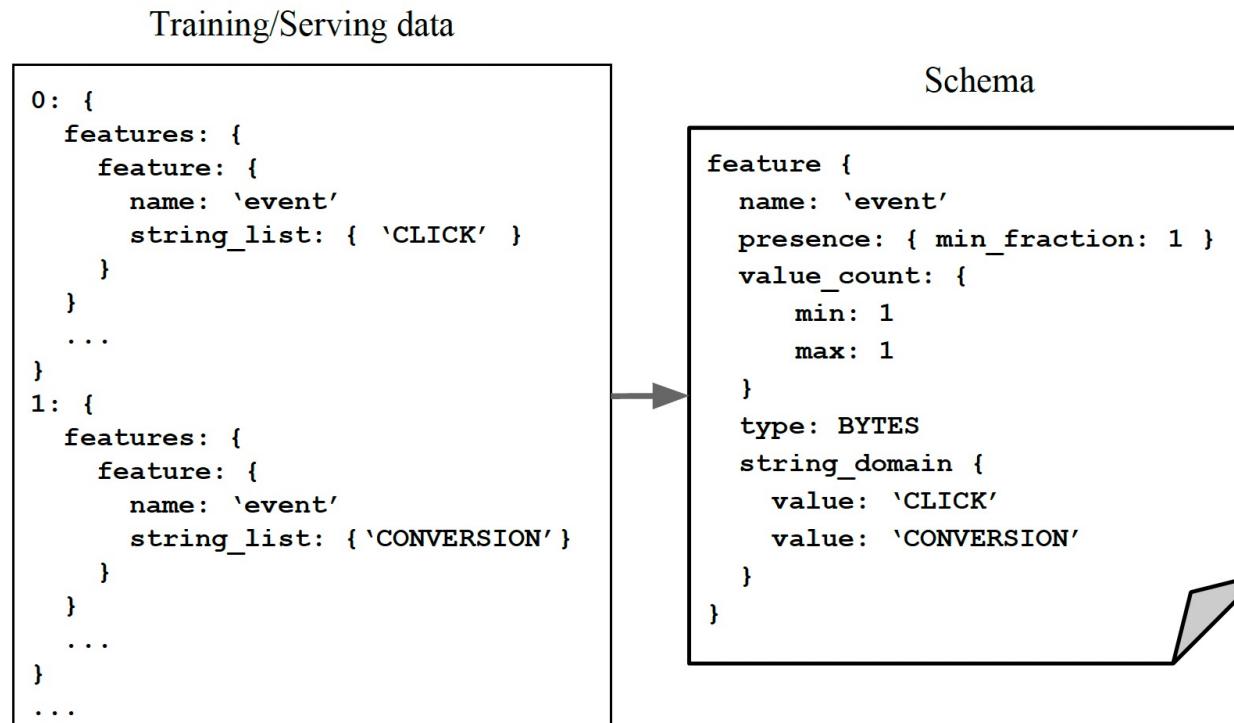
- Validation par lot unique
 - Y a-t-il des anomalies dans un seul lot de données ?
- Validation inter-lots
 - Y a-t-il des changements entre les données d'entraînement x service ?
 - Y a-t-il des changements entre les lots successifs de données ?
- Test du modèle
 - Y a-t-il des hypothèses dans le code de formation qui ne sont pas reflétées dans les données?

Validation par lot unique

- Hypothèses ou attentes:
 - Les caractéristiques des données doivent être stables au sein de chaque lot
 - Certaines caractéristiques restent stables entre les lots qui sont plus proches dans le temps
- Pour valider ces caractéristiques attendues, les auteurs utilisent la notion traditionnelle de **schéma**

Schema

- Coder les contraintes de données et la sémantique



Schema (cont'd)

- Lorsqu'une anomalie est détectée, le rapport inclut des suggestions de correction.
 - Corriger les données ou corriger le schéma

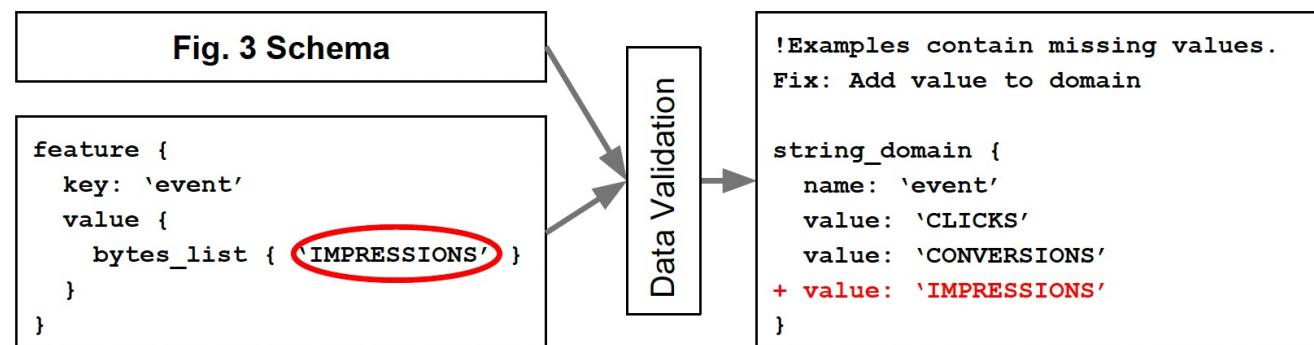


Figure 4: Schema-driven validation

Cycle de vie du schéma

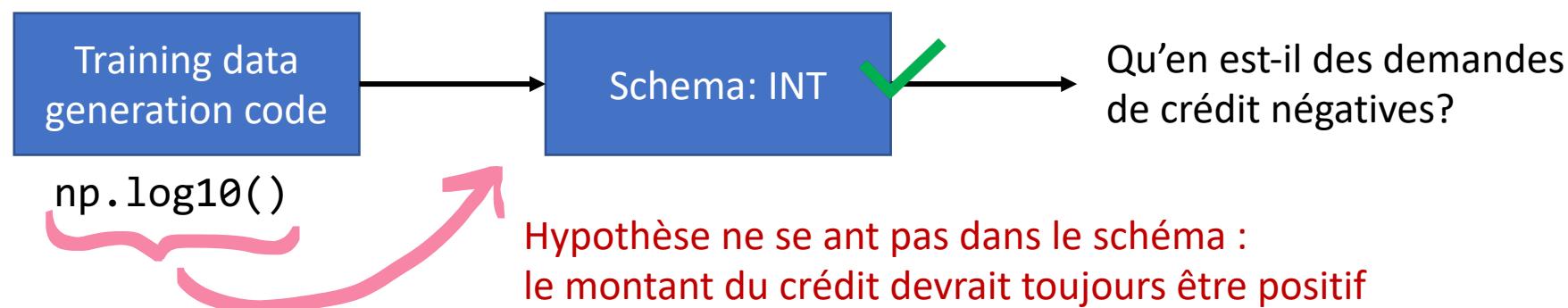
- Les propriétaires de pipelines sont responsables du schema
 - Le schéma sera constamment mis à jour et validé
- Le validateur de données génère une **structure initiale**
 - Toutes les attributs en fonction du lot initial
 - Fondé sur des « heuristiques raisonnables »

Validation inter-lots

- **Biais de service de formation:** training data vs serving data
- Biais de attributs: les valeurs sont différentes
- Biais de distribution: la distribution des valeurs est différente
- Biais de servir: seule une partie des données est servie
- Validation: L'utilisateur peut spécifier la tolérance des paramètres
 - Les changements dans les valeurs ou la distribution au-delà des niveaux de tolérance sont avertis

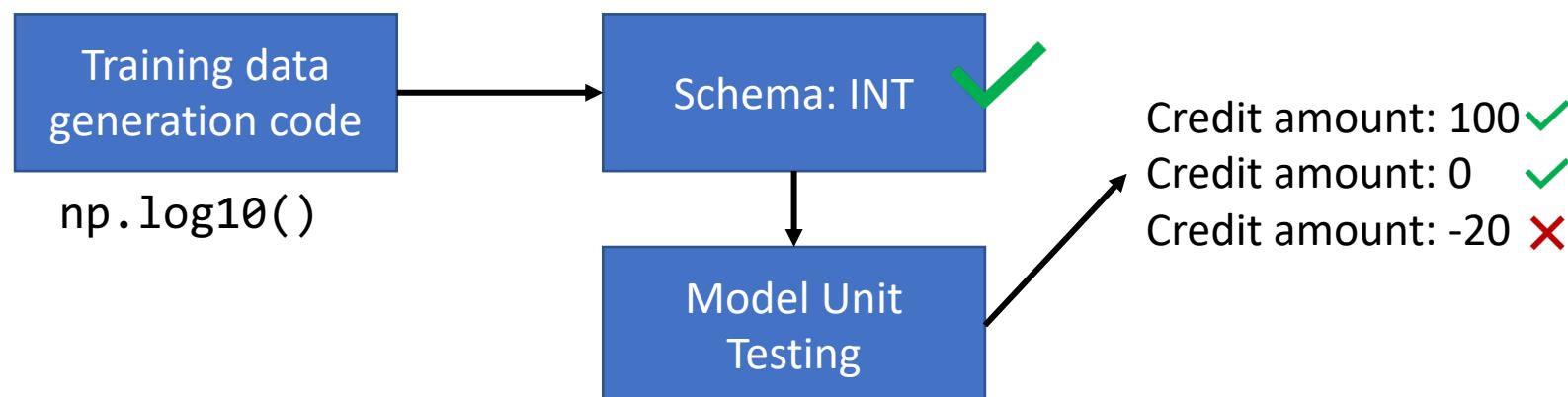
Test d'unités des modèles

- Utilise le schéma pour générer des tests à l'aide des tests Fuzz
 - Test de fuzz: génère des entrées aléatoires pour le test.
- Exemple tiré de le German credit report
 - Credit amount



Test d'unités des modèles (cont.)

- Utilise le schéma pour générer des tests à l'aide des tests Fuzz
 - Test de fuzz: génère des entrées aléatoires pour le test.
- Exemple tiré de le German credit report
 - Credit amount

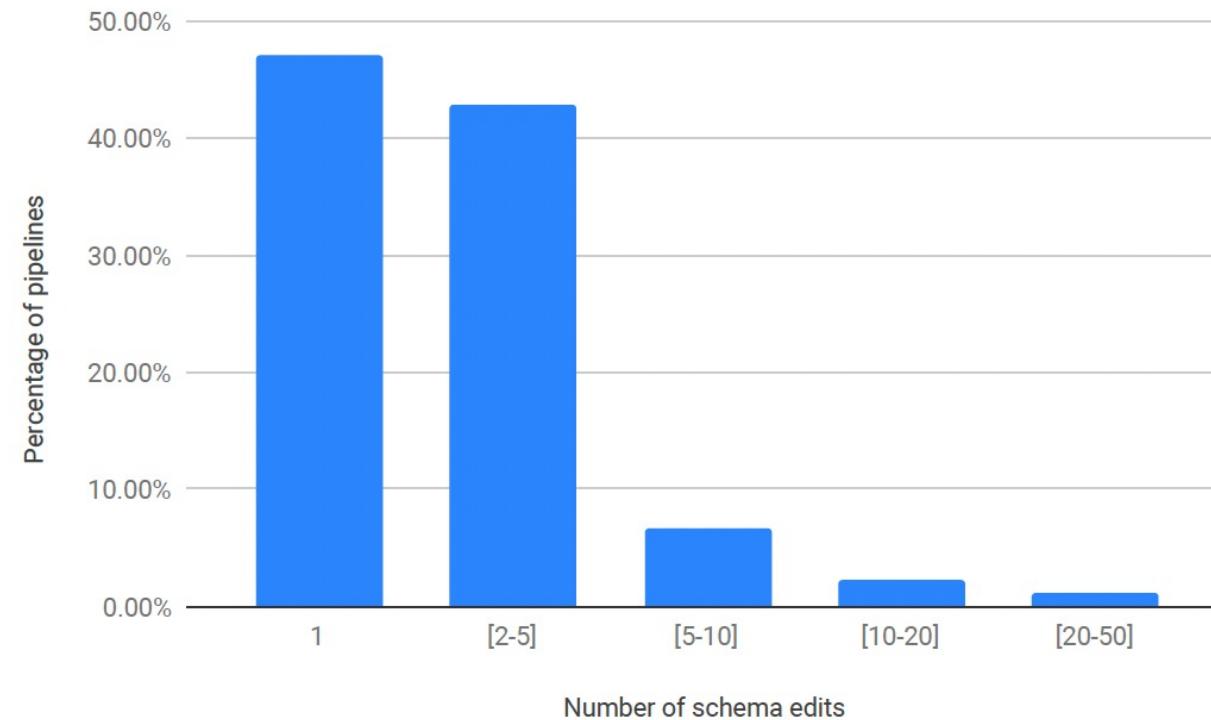


Validation empirique

- Déployé en production
 - Partie de la plateforme TFX
 - Plusieurs pétaoctets de données d'entraînement et de service par jour
- À quelle fréquence les utilisateurs modifient-ils leurs schémas ?
- Quelles sont les anomalies les plus courantes détectées?

À quelle fréquence les utilisateurs modifient-ils leurs schémas ?

Basé sur des pipelines de 700 ML



Anomalies les plus courantes détectées/corrigées

Anomaly Category	Used	Fired	Fixed given Fired
New feature column (in data but not in schema)	100%	10%	65%
Out of domain values for categorical features	45%	6%	66%
Missing feature column (in schema but not in data)	97%	6%	53%
The fraction of examples containing a feature is too small	97%	3%	82%
Too small feature value vector for example	98%	2%	56%
Too large feature value vector for example	98%	<1%	28%
Data completely missing	100%	3%	65%
Incorrect data type for feature values	98%	<1%	100%
Non-boolean value for boolean feature type	14%	<1%	100%
Out of domain values for numeric features	67%	1%	77%

Corriger si l'anomalie était n'est pas déclenché dans l'les 2 prochains jours.

Test d'unité du modèle

- Plus de 70 % des pipelines 1+ unité modèle définie
 - 80 mille tests exécutés par mois
- 6% des exécutions échouées
 - Le code de formation comportait des hypothèses incorrectes
 - OU
 - Le schéma était sous-spécifié

Études de cas

- Fonctionnalités manquantes dans le pipeline recommandé par Google Play
 - Attributs dans les données d'entraînement qui n'existe pas dans le service
 - Fix: +2% app install rate
- Débogage des données
 - Les attributs ont été **silencieusement supprimées** du code de génération de service
 - Fix: une meilleure parité de performance (training and serving)

Discussion ouverte



Chapitre 3 – Projet de cours

- Explorez votre jeu de données et écrivez vos observations.
 - Consultez le notebook + présentation de la semaine 2
- Décrivez chaque attribut:
 - Type de valeur (catégorie, numérique)
 - Distribution des valeurs
 - Problèmes potentiels (données manquantes, les biais)
- Décrire le traitement des données/l'ingénierie des fonctionnalités
 - Par attribut: Normalisation, scaling, transformation...
 - Sélection des fonctionnalités
 - Consultez l'article “The Data Linter”

Chapitre 3 – Projet de cours

Validation des données

Créer une **schema** pour toutes les fonctionnalités, notamment:

- Fréquence attendue dans l'ensemble de données
- Valeurs de plage attendues
- Consultez l'article » Data Validation for Machine Learning «
- Planifier un composant de validation qui:
 - Lit le schéma défini
 - Lit votre jeu de données
 - Valide le jeu de données

February 2023

< >

S	M	T	W	T	F	S
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	1	2	3	4



Présentation du Plan du Projet

Présentation du Plan du Projet – Mars 9

- Chaque groupe doit présenter pour **15 minutes**
- Contenu de 4 premier chapitre
 - Exigences
 - Architecture
 - Analyse et validation des données
 - Sélection du modèle
- Opportunité de soumettre une version du rapport (Moodle)
 - Jusqu'a Mars 9

References

- [\[Video\] Data Validation for Machine Learning](#)