Détection et correction des problèmes de qualité de données par apprentissage automatique

Detection and correction of data quality problems with machine learning

par Laure BERTI-ÉQUILLE

Directrice de Recherche

Institut de Recherche pour le Développement

ESPACE-DEV

Montpellier, France

Résumé (~500 - 800 signes) :

Ce dossier présente l'évolution récente des techniques d'évaluation et l'amélioration de la qualité des données basées sur des méthodes d'apprentissage automatique. Il décrit les solutions issues principalement du monde de la recherche ainsi que les principales approches mises en œuvre pour détecter et corriger de façon semi-automatique les principaux problèmes de qualité des données que sont les données aberrantes, incohérentes ou manquantes et les doublons.

Abstract (~500 – 800 signs) :

This synthesis presents the recent evolution of techniques for the evaluation and improvement of data quality in databases based on machine learning methods. It describes recent solutions proposed mainly by the academia as well as main approaches implemented to detect and correct main data quality problems such as outliying, inconsistent or missing data, and duplicates in a semi-automatic manner.

Mots-clés / *Keywords* :

Qualité des données, science des données, détection d'anomalies, nettoyage des données, apprentissage automatique, gestion de la qualité des données, métadonnées, indicateurs de qualité

Data quality, data science, anomaly detection, data cleaning, machine learning, data quality management, metadata, data quality indicators.

1. INTRODUCTION

Des progrès significatifs ont été accomplis ces dernières années dans la conception d'outils permettant d'automatiser l'évaluation, le suivi et l'amélioration de la qualité des données, notamment grâce aux avancées technologiques de l'Intelligence Artificielle, et en particulier, de l'apprentissage automatique à partir des données (*ML – Machine Learning*). Les techniques d'apprentissage ont été rendues opérationnelles à grande échelle et largement déployées dans tous les secteurs d'activités afin d'automatiser les tâches de prédiction et de classification pour l'aide à la décision dans de nombreux domaines d'application (santé, finance, marketing, etc.). La fiabilité des résultats de ces méthodes demeure cependant très dépendante de la qualité des données en entrée des modèles d'apprentissage. La qualité des données est rarement au rendez-vous et les données sont souvent imparfaites. Ainsi, deux approches complémentaires sont proposées : l'une émanant de la communauté de recherche en gestion des données visant à corriger les données en amont des chaînes d'analyse (par nettoyage ou réparation des données) et l'autre issue de la communauté des chercheurs et praticiens en apprentissage (*data scientists*) visant à développer des modèles plus robustes au bruit et et plus performants en mettant davantage l'accent sur la transformation et la préparation des données.

Pendant des décennies, pour la communauté spécialisée en gestion des données, le nettoyage des données a consisté à détecter les incohérences dans les bases de données relationnelles sous forme de violation de contraintes, à les « réparer » et à proposer des solutions souvent théoriques permettant le raisonnement à partir des données incohérentes, leur interrogation, la vérification et la satisfaction de contraintes d'intégrité [**], la découverte de dépendances fonctionnelles [**] ou de règles métier [**] dans le but de corriger la base en un nombre minimal de mises à jour [**], d'éliminer les doublons [**] ou de retourner une réponse cohérente aux requêtes [**].

Dans la pratique, les analystes confrontés à des anomalies dans leurs jeux de données utilisent, quant à eux, des chaines de prétraitement permettant de préparer et transformer les données pour qu'elles soient conformes aux attendus des modèles employés. Ils utilisent un ensemble de transformations automatiques et de procédures d'étiquetage souvent manuelles. En pratique, l'approche la plus courante consiste soit à exclure de l'analyse les données en erreur soit à les gérer séparément en utilisant souvent plusieurs méthodes pour leur détection et leur remplacement.

Dans cet état de l'art qui ne saurait être exhaustif, notre objectif est de montrer : 1) que les erreurs dans les données peuvent considérablement affecter les résultats des modèles d'apprentissage ; et 2) qu'il existe de nombreuses techniques d'apprentissage permettant détecter les anomalies et les corriger de façon semi- voire totalement automatique et nous en ferons un rapide tour d'horizon limité au cas des données structurées sous forme de tables.

Les perspectives de recherche et de développement sont nombreuses pour évaluer la qualité des données complexes, notamment multimodales et spatio-temporelles (combinant, par exemple, texte, image, audio, vidéo, série temporelle, et géolocalisation) car assez peu de travaux existent aujourd'hui pour combiner ces différents signaux, détecter et corriger les anomalies en croisant les modalités. La voie est ouverte car les techniques d'apprentissage automatique offrent de nombreux avantages pour capturer ces différents types de données par des représentations sous forme vectorielle (tenseurs, plongements (*embeddings*)) qui permettant ainsi de les analyser conjointement et d'exploiter des caractéristiques latentes dans les données.

2. IMPACTS DE LA QUALITE DES DONNEES EN APPRENTISSAGE AUTOMATIQUE

Les principaux types d'erreurs à considérer pour faire état de la qualité d'un jeu de données sont les valeurs manquantes, les valeurs aberrantes (*outliers*), les valeurs incohérentes (ne satisfaisant pas un ensemble de contraintes prédéfinies), et enfin les doublons, comme l'illustre le tableau 1.

Tableau 1. Principaux problèmes de qualité de données avec les principales méthodes de détection et de correction associées

Type de problème	**Méthode de détection**	**Méthode de correction**
Valeur manquante	Valeur vide, NaN, NULL	- Suppression de l'enregistrement complet ou de l'attribut s'il contient plus de 30% de valeurs manquantes dans son domaine de valeurs - Remplacement de la valeur manquante : • par la valeur catégorielle la plus fréquente ou par celle des plus proches voisins • remplacement par la moyenne/médiane/mode pour les valeurs numériques manquantes
Valeur aberrante ou hors norme	IQR, 3σ, ZScore, Isolation Forest, LOF, SVM, MCD, etc.	- Suppression de l'enregistrement complet - Ecrêtage de la valeur en utilisant un seuil minimal ou maximal - Remplacement par des méthodes d'imputation univariée ou multivariée
Valeur incohérente	OpenRefine (https://openrefine.org/) Détection de dépendances fonctionnes, contraintes de déni, contraintes d'intégrité, etc.	- Suppression de l'enregistrement complet - Remplacement d'une ou plusieurs valeurs de façon à satisfaire l'ensemble des contraintes du domaine
Doublons	ZeroER () (Getoor)	- Suppression d'un des enregistrements - Fusion des enregistrements
Erreur de label	CleanLab (https://cleanlab.ai/)	- Remplacement du label

Toutefois pour la communauté en apprentissage, les problèmes de qualité de données sont généralement englobés sans détail dans la notion de bruit associé d'une part, aux données, et d'autre part, au paramétrage du modèle. Dans ce contexte, il s'agit plutôt d'étudier et de quantifier l'incertitude prédictive selon les deux volets représentés dans la figure 1 que sont :

- **l'incertitude épistémique**, liée au choix optimal des paramètres du modèle d'apprentissage utilisé pour la tâche de prédiction. Ce type d'incertitude décroit lorsque la taille du jeu de données d'entrainement augmente ;
- **l'incertitude aléatoire**, liée aux données, qui comprend notamment pour la classification : les problèmes de recouvrement entre classes ou les erreurs dans les labels, ou bien pour la régression : la variance des erreurs qui peut être constante (homoscédasticité) ou non (hétéroscédasticité). Ce type d'incertitude ne peut être réduit en ajoutant davantage de données d'entraînement. Les efforts pour réduire ce type d'incertitude porteront alors soit sur la préparation et la transformation des données en amont soit sur des stratégies permettant de rendre le modèle plus robuste.

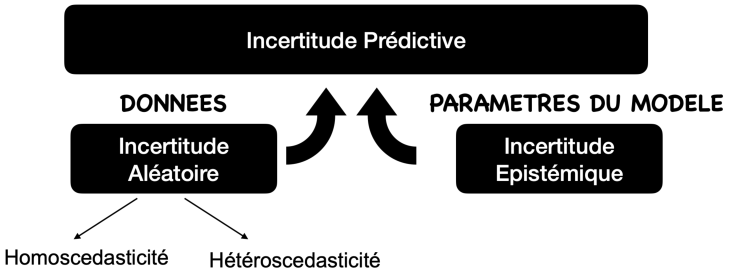

Figure 1. Incertitude de la prédiction par apprentissage

Tous les problèmes de qualité des données évoqués dans le tableau 1 sont donc ainsi englobés au sein de l'incertitude aléatoire bien que ceux-ci n'aient souvent rien d'aléatoire et surviennent au contraire de façon récurrente et/ou selon certains patterns ou conditions d'occurrence.

De nombreuses approches ont été proposées pour quantifier l'incertitude que ce soit pour améliorer la classification ou la prédiction à partir de données. Certains travaux se sont notamment intéressés à la robustesse des méthodes d'apprentissage face à des perturbations du jeu de données initial due à des attaques visant à corrompre les données (*data poisoning)* que ce soit lors de l'entrainement ou lors du test et de la validation du modèle. Dans ce dernier cas, les nouvelles données utilisées peuvent très différentes des caractéristiques des données sur lesquelles le modèle a été appris : elles sont dites hors distribution (*OOD – Out Of Distribution*) et très éloignées des données utilisées pour l'entrainement causant une dégradation des performances du modèle.

2.1. Impact en classification

L'exemple de la figure 2 illustre comment des données corrompues peuvent compromettre radicalement les résultats d'une classification en 2 classes [Biggio 17]. Chacun des 4 graphiques montre les données d'entraînement (points rouges et bleus pour chacune des deux classes) et la classification obtenue est représentée par une ligne droite noire séparatrice. La fraction de points corrompus injectés dans l'ensemble d'apprentissage est signalée au-dessus de chaque graphique, ainsi que l'erreur de test du classifieur (entre parenthèses). Dans cet exemple, un algorithme basé sur le gradient génère les points en anomalie pour l'attaque. Il procède en clonant des points existant initialement dans le jeu de données d'entraînement (indiqués par des croix blanches) et en changeant leur label. Les trajectoires du gradient (lignes noires en pointillée) sont ensuite suivies jusqu'aux optimas locaux pour obtenir les points finaux (points encerclés en noir) permettant de corrompre la classification à l'entrainement. Après entrainement du classifieur SVM linéaire, on observe un changement notable de l'angle de la ligne de démarcation entre les deux classes dès que sont introduits de 5%, 10% à 20% de points en anomalie à partir du jeu de données initial.

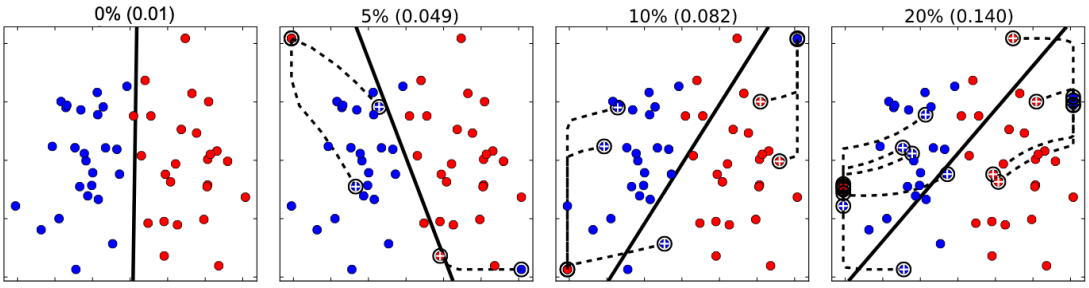

Figure 2. Exemples de corruption de données affectant le résultat d'une classification par SVM à noyau linéaire extrait de [*Biggio 2017*]

Cet exemple montre que seulement quelques points avec des labels incorrects dans le jeu de données d'entrainement peuvent, s'ils sont bien choisis, suffire pour modifier complètement les performances d'un classifieur et la fonction apprise. C'est précisément ce qui se passe et parfois de façon amplifiée lorsque les données des variables explicatives ou

les labels utilisés pour l'entraînement ou le test d'un modèle sont de faible qualité et comportent des erreurs.

2.2. Impact en régression

L'exemple présenté dans la figure 3 montre cette fois les prédictions obtenues par plusieurs méthodes de régression : Ridge (en vert) ou Huber (autres couleurs); celles-ci sont fortement influencées par les valeurs aberrantes présentes dans le jeu de données. Le régresseur de Huber est moins sensible aux valeurs aberrantes, car le modèle utilise une fonction de perte linéaire. Lorsque le paramètre epsilon est augmenté (de 1.0 à 1.9) pour le régresseur de Huber, la fonction de décision se rapproche de celle obtenue par la régression Ridge. Ceci montre bien l'impact considérable des problèmes de qualité de données, en l'occurrence des données aberrantes (ou *outliers*) sur le résultat de la régression qui peut être très variable (si l'on compare les inclinaisons de la droite représentant la fonction de décision). On notera que ces données erronées conditionnent non seulement le choix de la méthode de régression (que l'on voudra la plus robuste possible) mais également le choix de son paramétrage (ici epsilon).

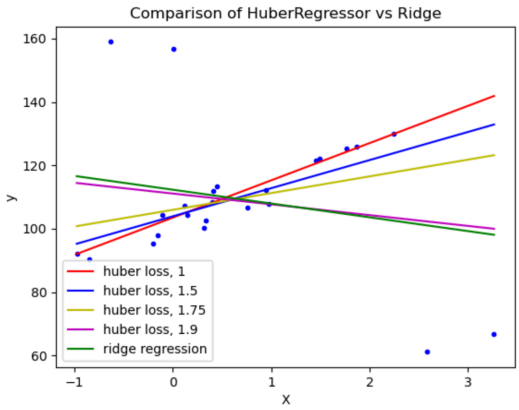

Figure 3. Exemple de régression à partir de données contenant des valeurs aberrantes (code disponible https://scikit-learn.org/stable/auto_examples/linear_model/plot_huber_vs_ridge.html#sphx-glr-auto-examples-linear-model-plot-huber-vs-ridge-py)

Dans le cas standard de données indépendantes et identiquement distribuées (i.i.d.), on peut pourra chercher à quantifier l'incertitude d'un résultat de régression de la façon suivante.

Pour des données d'apprentissage $(X, Y) = \{(x_1, y_1), \cdots (x_n, y_n)\}$ ayant une distribution inconnue notée $P_{X,Y}$, on peut supposer que :

$$Y = \mu(X) + \epsilon$$

avec μ, la fonction de régression que l'on cherche à déterminer pour prédire Y à partir de X et le bruit noté $\epsilon_i \sim P_{YIX}$.

On estimera l'intervalle d'incertitude de la prédiction par validation croisée pour un certain degré de confiance α en quatre étapes (BARBER):

1. On divise l'ensemble d'apprentissage en K sous-ensembles disjoints $S_1, S_2, ..., S_K$ de taille égale ;
2. K fonctions de régression $\hat{\mu}_{-S_k}$ sont ajustées sur l'ensemble d'apprentissage sans le k-ième pli ;
3. Le résidu hors pli R_i est calculé pour chaque i-ième point tel que $R_i = \left|Y_i - \hat{\mu}_{-S_{k(i)}}(X_i)\right|$ où $k(i)$ est le pli contenant i.
4. Les fonctions de régression $\hat{\mu}_{-S_{k(i)}}(X_i)$ sont ensuite utilisées pour estimer les intervalles de prédiction selon un certain degré de confiance α. L'intervalle de

prédiction et donc l'incertitude associée à la prédiction d'un nouveau point X_{n+1} est calculé à partir des différents résidus de ces fonctions tel que :

$$\hat{\mu}(X_{n+1}) \pm \left((1-\alpha)\ quantile\ de\ |Y_1 - \hat{\mu}_{-1}(X_1)|, \ldots, |Y_n - \hat{\mu}_{-n}(X_n)|\right).$$

Différentes stratégies peuvent être ainsi mises en place pour estimer le biais des résultats dû à l'imperfection des données ou du modèle. Par la suite, il s'agit d'opter pour des approches visant à rendre la prédiction plus robuste (BELKIN)(STEINHARDT), notamment par des techniques d'assemblage de plusieurs modèles (*ensembling*), de régularisation (BISHOP), d'augmentation de données (Van DYK), ou de drop-out (pour les modèles d'apprentissage profonds).

Cependant, toutes ces approches nécessitent une relativement grande expertise en modélisation prédictive. Plusieurs bibliothèques open source disponibles (telles que Dataprep https://github.com/sfu-db/dataprep) et Facets (https://github.com/PAIR-code/facets) combinant la visualisation et des méthodes de détection peuvent faciliter la préparation des données pour l'apprentissage.

3. DETECTION ET CORRECTION PAR APPRENTISSAGE AUTOMATIQUE

Des travaux récents ont pu montrer que les modèles d'apprentissage automatique peuvent être utilisés avec précision pour identifier les problèmes dans les données et corriger certains types d'erreurs avec des mécanismes de correction complexes qui étaient jusqu'ici réalisés manuellement ou basés sur des heuristiques difficiles à maintenir. De nouvelles stratégies d'apprentissage peuvent également déterminer quelles sont les corrections nécessaires selon l'objectif d'analyse car il n'est pas forcément nécessaire (ni même faisable) de corriger tous les problèmes.
Les approches basées sur l'apprentissage sont utilisées à la fois pour la détection et la correction des données erronées. Elles reposent sur des paires d'enregistrements erronés et corrects utilisées comme exemples pour entraîner le modèle d'apprentissage. Mais la conception de modèles suffisamment expressifs et donc complexes nécessite beaucoup de données d'entrainement. Selon la tâche (détection de valeurs aberrantes, doublons, incohérences, etc.), la constitution de ces jeux de données d'apprentissage peut s'avérer très difficile car l'annotation est souvent manuelle. Les stratégies semi-supervisées ou faiblement supervisées peuvent cependant éviter le recours à un trop grand nombre d'exemples annotés manuellement. Dans la suite de cet article, nous présenterons plusieurs approches existantes de façon couvrir le spectre des méthodes allant de celles qui nécessitent l'intervention de l'utilisateur jusqu'à celles purement automatiques.

3.1. Détection des anomalies par apprentissage

Les erreurs étant généralement des événements rares, les modèles de détection se retrouvent confrontés à plusieurs problèmes, notamment : (1) le déséquilibre des classes ; les données normales sont majoritaires alors que les anomalies sont rares ; (2) un jeu d'entrainement limité peut présenté des problèmes de significativité et de représentativité des erreurs possibles dans les données ; (3) les méthodes de détection ne capturent pas forcément les mêmes erreurs selon leur sensibilité et/ou leur paramétrage, par conséquent, elles auront souvent des résultats contradictoires parfois difficiles à unifier.

Comme le montre la figure 4, les différentes méthodes de détection peuvent être combinées pour améliorer la détection les erreurs.

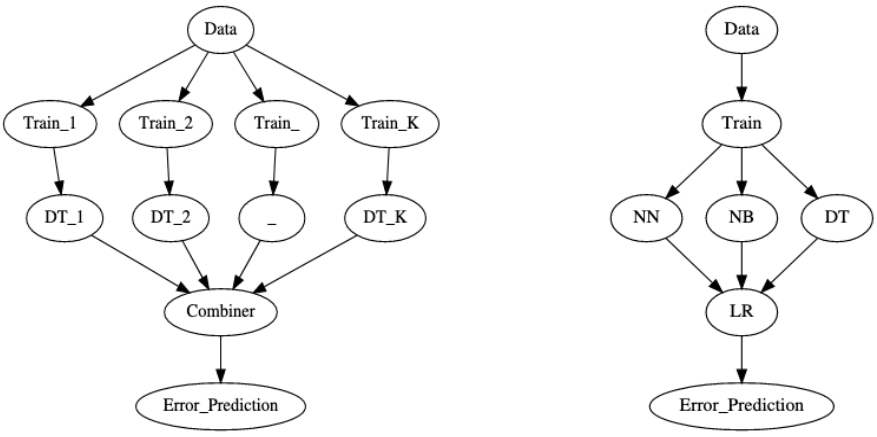

Figure 4. Deux stratégies principales combinant plusieurs methods de détection d'erreurs ( )

L'approche d'assemblage consiste ainsi à combiner plusieurs méthodes de détection d'erreurs à partir des données d'entraînement (Train) en utilisant : (a) l'agrégation de modèles de même type tels que les arbres de décision (DT) ou (b) l'empilement de modèles hétérogènes tels que des Réseaux de neurones (NN), des modèles Bayésiens Naïfs (NB) et des méthodes de régression logistique (LR).

HoloDetect (HEIDARI et al., 2019) utilise l'augmentation des données en apprenant le modèle des erreurs observées et en appliquant un modèle qui génère des erreurs de façon synthétique. ActiveL est une autre version de HoloDetect qui utilise l'apprentissage actif choisissant précisément les exemples que l'utilisateur aura à annoter pour réduire l'effort d'annotation préalable à l'apprentissage.

Dans le système Rain (Wu et al., 2020), l'utilisateur peut annoter les erreurs (appelées "réclamations") dans les résultats d'une requête et indiquer si le résultat agrégé de la requête est conforme à une contrainte particulière. Par exemple, l'utilisateur peut spécifier que le nombre des ventes en mars doit être supérieure à 40 ou que toutes les valeurs sont trop faibles. Rain classe alors les enregistrements erronés (et leur correction possible) dans l'ensemble des données d'apprentissage en fonction de la résolution des réclamations et estime le gradient du résultat de la requête par rapport aux modifications apportées dans l'ensemble de données d'entraînement (par suppression d'un enregistrement ou ajout d'un nouvel enregistrement corrigé). Une seule annotation d'un résultat agrégé en sortie du modèle permet d'identifier les erreurs dans le jeu de données d'entraînement plus efficacement que d'étiqueter manuellement plusieurs centaines d'erreurs en entrée du modèle.

Depuis plusieurs décennies, de nombreux travaux se sont intéressés à la détection des données aberrantes (*outliers*) avec des approches statistiques, basées sur des notions de distance ou de densité, des approches probabilistes ou plus récemment des approches par apprentissage profond avec des réseaux de neurones. La figure 5 illustre 12 méthodes de détection de valeurs aberrantes appliquées à un même jeu de 200 points dont 40 anomalies représentées par des points noirs. Pour chaque méthode, on observe en rouge la fonction de décision permettant de discerner les données « normales » des anomalies et la légende nous indique également le nombre d'erreurs commises dans la méthode de détection. Selon la distribution du jeu de données et la complexité des anomalies, les méthodes seront plus ou moins efficaces et performantes, avec souvent en tête Isolation Forest (), SVM () et LOF ().

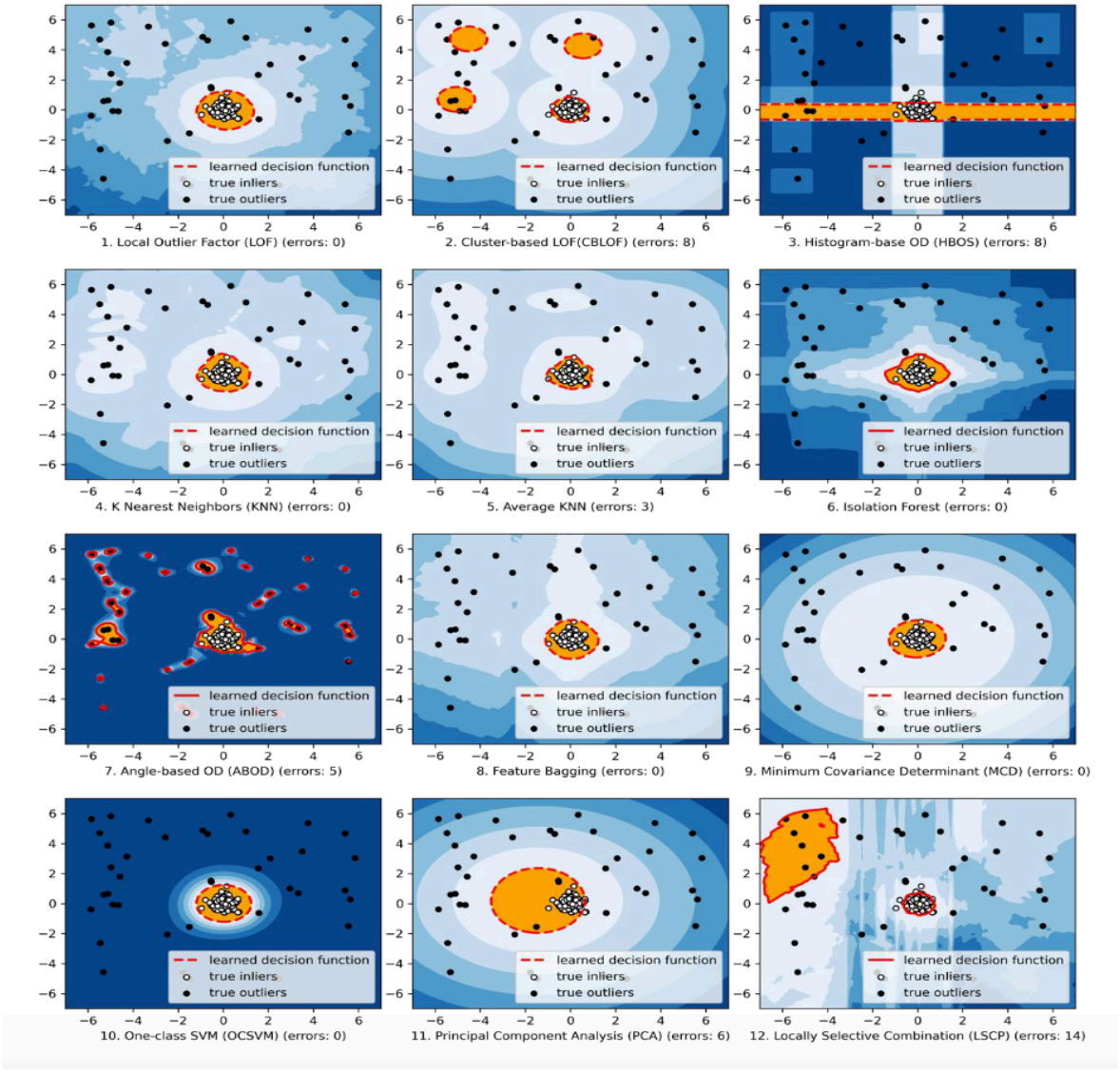

Figure 5. Détection de valeurs aberrantes par 12 méthodes de la littérature (survey)

3.2. Correction des erreurs par apprentissage

Les systèmes de réparation des données tels que Baran (MAHDAVI 2020) et HoloClean (rekatsinas 2017) combinent différents types de signaux pour corriger plus précisément les erreurs. Baran (MAHDAVI 2020) est un système qui combine plusieurs stratégies de correction d'erreurs et utilise l'apprentissage actif pour former un modèle qui prédit la stratégie de réparation à utiliser. La bibliothèque de stratégies associée est extensible et peut inclure de nouveaux modèles prédictifs.

ActiveClean (KRISHNAN et al, 2016) est un système de nettoyage des données utilisant des modèles d'apprentissage basés sur des fonctions de perte convexes. Il traite un modèle appris à partir des données d'entrainement ayant des erreurs comme un point sous-optimal le long de la fonction de perte par rapport à un ensemble de données d'apprentissage parfait (qui reste inconnu). Il traite ensuite le nettoyage des données comme un problème de descente de gradient stochastique (SGD), où, à chaque étape, le système échantillonne et demande à l'utilisateur de corriger des enregistrements, ce qui a pour conséquence de déplacer le modèle le long du gradient de plus grande pente. L'utilisation des modèles convexes offre au système des garanties de convergence. ActiveClean choisit simplement quelles sont les données d'entrainement qui doivent être corrigées d'une itération à l'autre, et compte sur l'utilisateur pour effectuer la correction proprement dite.

CPClean (KARLAS 2021) nettoie progressivement le jeu de données d'apprentissage jusqu'à ce qu'il soit certain qu'aucune autre correction ne puisse être effectuée si elle modifie les prédictions du modèle. Pour cela, il propose le concept de prédictions certaines (*certain prediction - CP*) pour un classifieur de type plus proches voisins. Une donnée de test est « prédite avec certitude » si tous les classifieurs formés sur toutes les corrections possibles du jeu de données d'entraînement donnent la même prédiction. CPClean calcule le nombre de modèles possibles pour la classification d'une donnée de test spécifique, et l'utilise ensuite pour quantifier l'impact de la correction d'une erreur ou d'une valeur manquante. Ainsi, étant donné un ensemble de données de validation, CPClean cherche à corriger uniquement les données d'apprentissage nécessaires pour que toutes les données de validation puissent être prédites avec certitude. CPClean compte néanmoins sur l'utilisateur pour réaliser les corrections. Il est spécialement conçu pour les modèles de classification par plus proches voisins et tire partie de leur robustesse face à de petites perturbations dans le jeu de données d'entraînement.

BoostClean (Krishnan 2017 2021) est spécialisé dans le cas où une erreur peut être spécifiée sur les données d'entraînement ou de test à l'aide d'un prédicat simple. A partir d'une bibliothèque prédéfinie de fonctions de détection et de nettoyage d'erreurs, il sélectionne automatiquement une séquence d'opérations de nettoyage qui maximisera la précision du modèle sur les données de validation (ou d'entraînement). À chaque étape, BoostClean utilise une stratégie de boosting statistique pour choisir une paire de fonctions de détection et de nettoyage ; il configure leurs paramètres et les applique à l'ensemble des données pour dériver une nouvelle itération du modèle plus performante que la précédente. BoostClean diffère des systèmes de nettoyage cités précédemment sur le point que les utilisateurs n'ont pas besoin de corriger manuellement les données. A la place, les utilisateurs spécifient simplement les fonctions de détection et de réparation appropriées pour leur domaine.

AlphaClean (KRISHNAN 2019) est un système très similaire à BoostClean car il génère une séquence d'opérations de nettoyage à partir d'une bibliothèque de fonctions de nettoyage. Mais au lieu au d'utiliser le boosting, le système utilise une recherche arborescente parallélisée et apprend une heuristique d'élagage pour réduire l'espace de recherche pour sélectionner les opérations de correction. AlphaClean utilise une procédure de recherche générique ce qui laisse l'utilisateur libre de définir sa propre fonction-objectif à optimiser (par exemple, la précision du modèle, ou le nombre de violations de contraintes, ou une combinaison des deux).

AutoSklearn (Feurer et al., 2015) utilise de la même manière la précision du modèle pour générer un pipeline d'apprentissage de bout en bout sans intervention humaine. Le pipeline comprend notamment des opérations de prétraitement des données et la sélection des hyper-paramètres du modèle. Pour restreindre l'espace de recherche, les opérations de prétraitement des données sont limitées à l'imputation de valeurs manquantes basée sur la moyenne, la médiane ou les valeurs les plus fréquentes, et sont appliquées à l'ensemble de données, soit pas du tout (par exemple, sans condition de correction).

SCARE (YAKOUT 2013) est l'un des tout premiers systèmes automatique utilisant plusieurs classifieurs pour réparer les données incohérentes selon un principe de minimalité, c'est-à-dire avec un nombre minimum de mises à jour de la base de données relationnelles. Il emploie plusieurs classifieurs probabilistes (par exemple, Bayésiens naïfs) entrainés sur des portions du jeu de données considérées comme fiables afin de produire une distribution de probabilités sur les valeurs des attributs qui sont considérés non fiables c'est-à-dire ayant des valeurs erronées. La distribution des probabilités conditionnelles a posteriori $\mathrm{P}(F \mid R)$ permet d'analyser les dépendances entre l'ensemble des attributs fiables noté R et l'ensemble des attributs non fiables noté F. Etant donné un tuple t composé de valeurs issues de ces deux ensembles, un premier classificateur $M1$ est utilisé pour prédire la valeur correcte d'un des premiers attributs les moins fiables $F1$ dans F, c'est-à-dire prédire $M1(F1)$ à partir des valeurs des attributs fiables pour ce tuple. Ensuite, un second classifieur $M2$ prédit la valeur du second attribut non fiable $F2$ à partir de $M1(F1)$ et de R. En procédant ainsi

autant de fois qu'il y a d'attributs non fiables, et connaissant les valeurs des attributs fiables R, chaque classifieur Mi prédit une valeur corrective possible. Le principe est de traiter chaque tuple de la base de données individuellement et d'utiliser l'ensemble de réparations candidates pour construire un graphe, pour lequel chaque sommet est une valeur d'attribut possible ; une arête est ajoutée entre une paire de sommets si les valeurs correspondantes coexistent dans d'autres prédictions. Les arêtes auront un poids dérivé des probabilités de prédiction obtenues. Le problème est alors modélisé comme un problème d'optimisation de graphe où il s'agit de trouver le sous-graphe avec le poids le plus élevé dans le graphe K-partite pour trouver la correction qui minimise le nombre de mise à jour et qui est corroborées par le plus de modèles.

Learn2Clean (BERTI-EQUILLE) est un système permettant de choisir la chaine de prétraitement et de nettoyage des données optimale par Q-Learning, une technique d'apprentissage par renforcement sans modèle. Il sélectionne :

- pour un ensemble de données en entrée ;
- un modèle d'apprentissage : par exemple CART, LDA ou NB pour la classification ; LASSO, OLS ou MARS pour la régression ; KMeans ou HCA pour le clustering ;
- et une métrique de performance du modèle : par exemple, précision pour la classification, MSE pour la régression ou Silhouette pour le clustering

la séquence optimale de tâches pour préparer les données de sorte que la métrique de qualité du modèle soit optimisée. Une bibliothèque de fonctions pour la préparation des données constitue l'espace de recherche qu'explore le système pour trouver la séquence de tâche optimale. Comme le montre la figure 6, elle comprend par exemple, des fonctions (ou actions à réaliser) pour la normalisation des données par MinMax (MM), Zscore (ZS) ou mise à l'échelle décimale (DS), la sélection d'attributs, l'imputation (par les plus proches voisins (KNN), MICE, KNN, les valeurs les plus fréquentes (MF)), la déduplication (exacte (ED) ou approximative (AD)), la détection de valeurs aberrantes (par écart interquartile (IQR), LOF ou Zscore) et la détection d'incohérences basée sur des contraintes (CC) ou des patterns (PC).

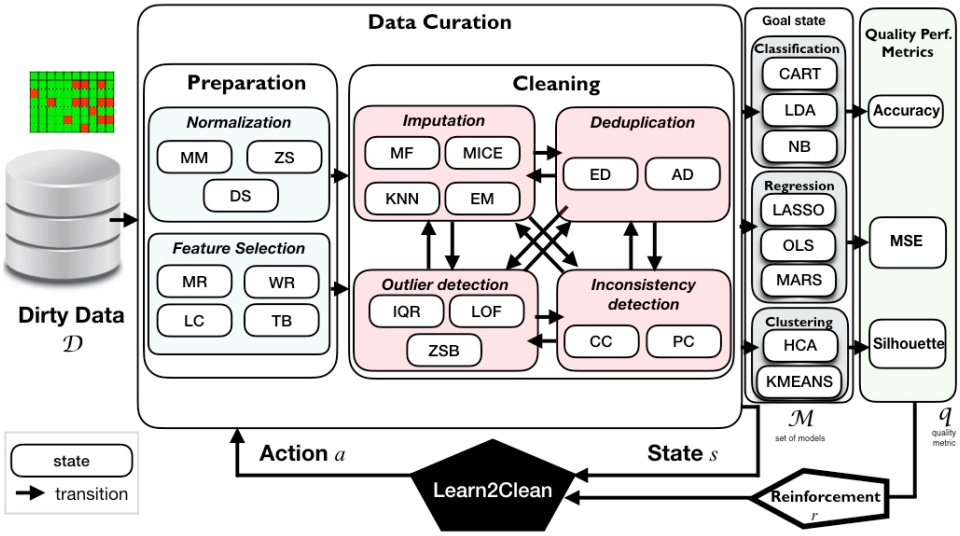

Figure 6. Architecture de Learn2Clean

En utilisant Q-learning, Learn2clean apprend par expérience de manière non supervisée. Chaque exploration équivaut à une séance d'entraînement qui consiste à passer de l'état initial (une des tâches de préparation ou de nettoyage représenté par des rectangles dans la figure) noté s, jusqu'à l'état final, résultat d'une classification, d'une régression ou d'un clustering. Lors de chaque session d'entraînement, le système explore l'espace des tâches reçoit une récompense si cela améliore la métrique de qualité choisie et met à jour la Q-table de la façon suivante (Equation (1)) :

Q-table

$$Q^{\pi}(s,a) \leftarrow (1-\alpha)\,.\,Q(s,a) + \alpha\,.\,\overbrace{\left(R(s,a) + \gamma\,.\,\max_{a'} Q(s',a')\right)}^{\text{valeur apprise}}$$

nouvelle valeur — taux d'apprentissage — ancienne valeur — récompense — facteur de remise — nouvelle valeur maximale

Equation (1)

Learn2clean utilise une politique π définie comme une fonction Softmax permettant la sélection de la prochaine action ; les actions *a* sont pondérées en fonction de l'estimation de leurs valeurs *Q(s,a)* à partir de l'état *s* dans la Q-table et elles sont échantillonnées à partir d'une distribution de Boltzmann (voire Equation 2). La probabilité de sélectionner l'action *a* ayant la Q valeur la plus élevée augmente avec le temps et le paramètre *k* qui contrôle la probabilité de sélectionner des actions non optimales. Si *k* est grand, toutes les actions seront sélectionnées uniformément. Si *k* est proche de zéro, la meilleure action (ayant la meilleure récompense) sera toujours choisie.

$$\pi = P(a \mid s) = \frac{e^{Q(s,a)/k}}{\sum_j e^{Q(s,a_j)/k}} \qquad \text{Equation (2)}$$

Comme l'a démontré Learn2Clean (), l'apprentissage par renforcement offre des perspectives très prometteuses pour automatiser l'exploration non seulement des données mais aussi des stratégies de nettoyage et de préparation des données.

Tableau 2. Principaux prototypes de recherche basés sur l'apprentissage automatique pour corriger les données structurées

Principaux Systèmes/Prototypes de recherche	**Techniques d'apprentissage**	**Spécificités de la correction/préparation des données**
Febrl [Churches et al., 2002]	Modèle de Markov Caché et Estimation du Maximum de Vraisemblance	Standardisation de données structurées (par exemple, en nom/adresse) sur la base du modèle probabiliste appris à partir de données d'entrainement
SCARE [Yakout, Berti-Equille, Elmagarmid, SIGMOD'13]	Multiple modèles d'apprentissage utilisés pour capturer les dépendances de données entre plusieurs partitions de données	Trouver la correction de données erronée qui maximise le bénéfice probable pour un certain seuil de coût de mise à jour 'une base de données relationnelle en tirant partie des dépendances fonctionnelles découvertes à partir d'attributs fiables
Continuous Cleaning [Volkovs et al., ICDE'14]	Classifieurs logistiques	Apprentissage à partir des préférences de correction des utilisateurs précédemment réalisées pour recommander les prochaines corrections de façon plus précise
Lens [Yang et al., VLDB'15]	Multiple modèles d'apprentissage utilisés pour encoder les contraintes d'un domaine	ETL déclaratif à la demande avec une hiérarchisation des tâches de nettoyage selon le résultat de requêtes probabilistes
HoloClean [Rekatsinas et al., VLDB 2017]	Inférence probabiliste à partir de graphes de facteurs par descente de gradient stochastique et échantillonnage de Gibbs	Découverte par apprentissage de règles statistiques et logiques, de contraintes de déni, et autres types de dépendances permettant d'établir des règles de correction des données qui passent à l'échelle
BoostClean [Krishnan et al., 2017]	AdaBoost	Découverte par apprentissage de règles statistiques et logiques et contraintes du domaine pour la détection et la correction des erreurs en maximisant la précision prédictive sur les données d'entrainement ou de test
Learn2Clean [Berti-Equille, 2019]	Anprentissage par renforcement Q-learning	Sélection de la séquence de tâche de nettoyage et de préparation qui optimise la métrique choisie selon la tâche (classification, régression ou clustering)

3.3. Application à la déduplication

La déduplication consiste à identifier et éliminer les enregistrements présents en plusieurs exemplaires au sein d'une base de données structurées de façon à n'en garder qu'un seul exemplaire. Le principe général illustré en Figure 7 peut s'appliquer sur le produit cartésien de deux ou plusieurs tables à fusionner ou au sein d'une seule et même table. Il est le suivant :

- Après une étape de nettoyage et de standardisation des valeurs d'un ensemble d'attributs choisis, un méthode dite de *blocking* est généralement appliquée pour réduire l'espace de recherche à un sous-ensemble (ou fenêtre) d'enregistrements à comparer ;
- Ensuite, une distance de similarité est calculée pour chaque paire d'enregistrements selon un ensemble de fonctions (comme par exemple, N-grams, Edit distance, Jaro, Soundex, etc.). Le résultat de chacune de ces fonctions peut être pondéré et agrégé à ce stade pour refléter au mieux la similarité entre les enregistrements deux-à-deux.
- Enfin, un modèle de décision détermine si les enregistrements sont suffisamment similaires pour être considérés comme doublons, ou non, ou encore comme doublons potentiels.

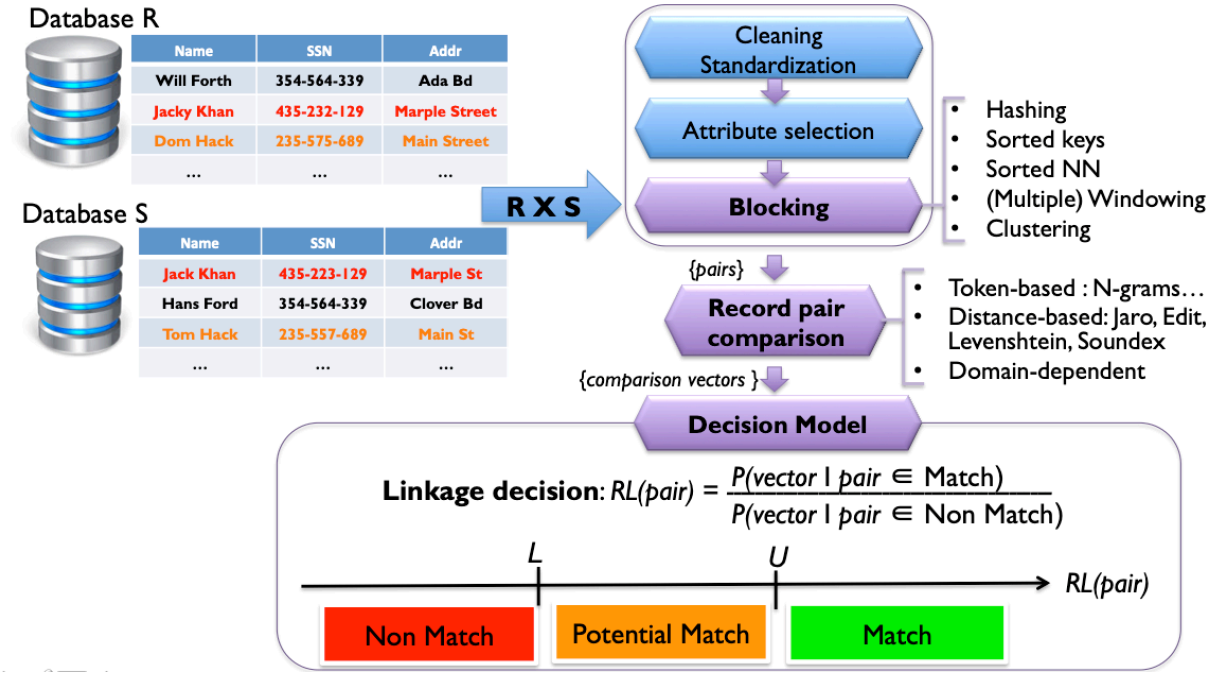

Figure 7. Approche générique de détection des doublons (figure à traduire)

De très nombreux systèmes ont été proposés pour la déduplication (par exemple, les prototype de recherche open source : Febrl (http://users.cecs.anu.edu.au/~Peter.Christen/Febrl/febrl-0.3/febrldoc-0.3/manual.html), Dedoop (https://dbs.uni-leipzig.de/dedoop), ou Nadeef (https://github.com/daqcri/NADEEF) ou les systèmes commerciaux souvent étendus au profilage de la qualité des données (tels que IBM DataStage (https://www.ibm.com/products/datastage), Data Ladder (https://dataladder.com/), Informatica DQ (https://www.informatica.com/products/data-quality/informatica-data-quality.html), Oracle EDQ (https://docs.oracle.com/middleware/1213/edq/DQARC/overview.htm#DQARC111), pour n'en citer que quelques-uns).

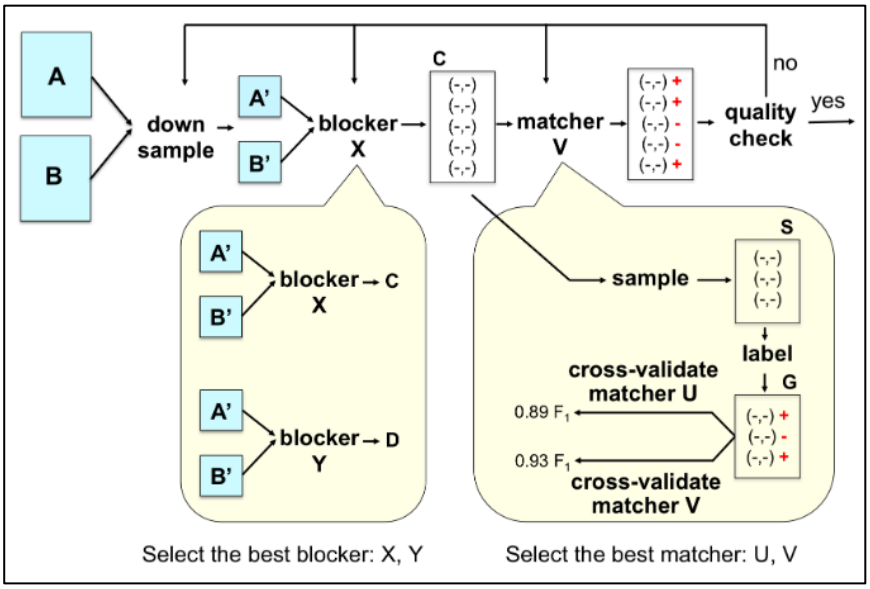

Figure 8. Fonctionnement du système Magellan (figure à traduire)

Le système Magellan (KONDA) dont le fonctionnement est illustré dans la figure 8 permet une appariement d'entités (*entity matching*) pour éliminer les doublons. Supposons que l'utilisateur U veuille faire fusionner deux tables A et B, chacune ayant 1 million de tuples. Essayer de trouver une séquence précise d'opérations sur ces deux tables prendrait trop de temps, car elles sont trop volumineuses. Par conséquent, U va d'abord sous-échantillonner les deux tables pour obtenir deux tables de taille plus raisonnable A' et B', chacune ayant par exemple, 100K tuples. Supposons que le système fournisse deux fonctions de *blocking* X et Y permettant de réduire l'espace de recherche. L'utilisateur peut ensuite expérimenter et choisir l'une des deux fonctions à appliquer sur les tables A′ et B′ pour obtenir un sous-ensemble de paires de tuples candidates C. Ensuite, l'utilisateur peut prélever un échantillon S à partir de C et étiqueter les paires d'enregistrement en double dans S avec le label « match » ou « no-match » représenté par "+" ou "-" sur la figure. A partir du sous-ensemble étiqueté G, le système propose alors deux algorithmes d'appariement (*matchers*) U et V basés sur l'apprentissage des labels fournis (par exemple, arbre de décision ou régression logistique). Ensuite, l'utilisateur peut utiliser l'ensemble étiqueté G pour effectuer une validation croisée pour U et V. Dans le cas où V a une précision plus élevée (F1 score de 0,93 dans la figure), l'utilisateur va sélectionner V et l'appliquer à l'ensemble C pour prédire l'appariement entre les paires d'enregistrements en tant que "match" ou "no-match". Enfin, l'utilisateur pourra effectuer un contrôle de qualité (en examinant, par exemple, un échantillon des prédictions), puis revenir en arrière, déboguer, modifier les étapes précédentes le cas échéant en remplaçant certains des composants (*blocker*/*matcher*).

D'un côté, engager efficacement l'humain dans la boucle de détection et de correction des erreurs est toujours difficile mais c'est souvent la seule voie permettant d'atteindre ou de garantir une certaine précision. D'un autre côté, automatiser les traitements et échantillonner les données restent les seules alternatives permettant de passer à l'échelle. Le compromis et l'équilibre entre ces deux voies restent bien évidemment à trouver selon le domaine d'application, la complexité des données et de leurs erreurs, les contraintes du domaine, et le budget alloué (en temps et personnel).

Plusieurs projets aussi bien académiques (DeepMatcher (https://github.com/anhaidgroup/deepmatcher), ZeroER (https://github.com/chu-data-lab/zeroer)) et industriels (Tamr (https://www.tamr.com/); Amperity (https://amperity.com/)) ont revisité le problème de la déduplication et utilisent l'apprentissage automatique.

En particulier, ZeroER (WU) montre comment utiliser l'approche par faible supervision pour réduire la quantité de données étiquetées nécessaires à l'apprentissage des modèles de déduplication.

Comme le suggère la figure 7, quels que soient les problèmes de qualité à détecter et corriger, de plus en plus de travaux sont attendus pour intégrer l'utilisateur davantage et de façon plus transparente et plus aisée afin qu'il puisse guider l'exploration et la sélection des stratégies de correction grâce ses connaissances a priori des données ou des méthodes, ses préférences et les besoins particuliers de son domaine d'application.

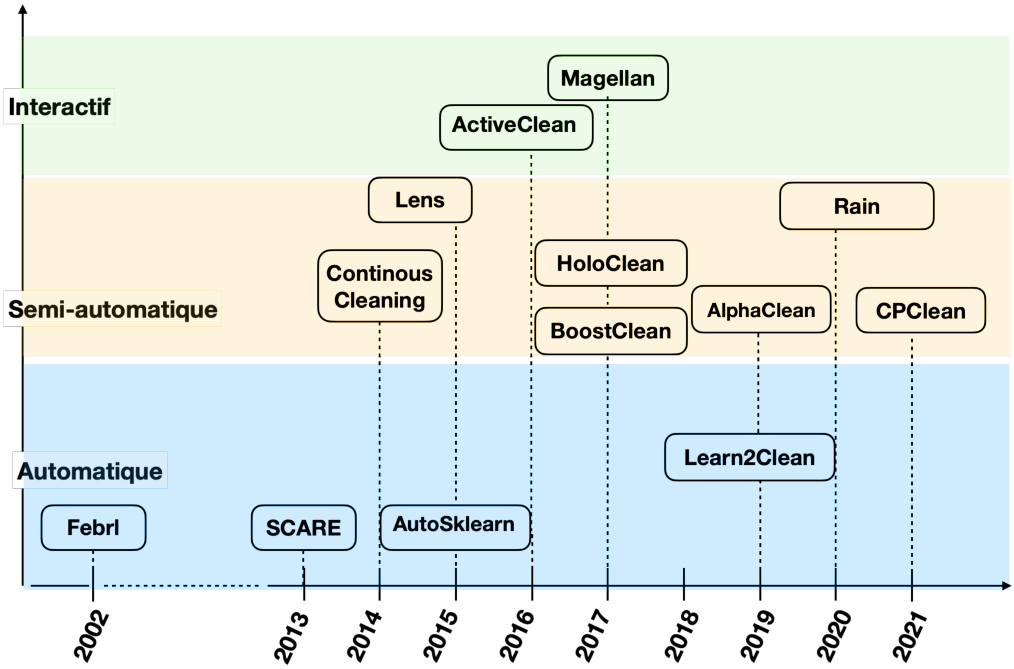

Figure 7. Chronologie des principaux systèmes de correction des données issus du monde de la recherche selon le mode d'implication de l'utilisateur

5. CONCLUSION

Le nettoyage et la préparation des données visant à en améliorer la qualité des données est largement considérée comme un élément essentiel, préalable l'application des techniques d'apprentissage automatique, car les erreurs dans les données impactent directement les performances des modèles prédictifs et la validité de leurs résultats. Traditionnellement, les solutions issues de la recherche pour le nettoyage des données se concentraient sur la correction des problèmes de qualité « dans l'absolu », parfois indépendamment de l'application utilisant les données. Dans un contexte industriel, l'application des techniques de nettoyage de données est souvent déterminée par des exigences de réduction de coûts associés aux données de mauvaise qualité ou de gain en performance d'après des indicateurs mesurables (*KPI - Key Performance Indicators*). Depuis l'essor de l'intelligence artificielle et la démocratisation de l'application des méthodes d'apprentissage, le nettoyage des données est désormais associée à l'étape de transformation et de préparation des données pour que celles-ci se conforment au mieux aux attendus des modèles et permettent d'en améliorer les performances (en termes de F1, précision, par exemple). Les techniques d'apprentissage peuvent être mises œuvre pour automatiser à grande échelle la correction des données. Cet article a présenté différentes approches de détection des anomalies et de nettoyage basées sur des méthodes d'apprentissage automatique. Les approches traditionnelles de nettoyage et de gestion de la qualité des données se sont jusqu'ici principalement concentrées sur les données structurées et textuelles. Les défis sont désormais de pouvoir évaluer et améliorer la qualité des données combinant plusieurs modalités (texte, image, vidéo, audio, et géolocalisation). Les modèles d'apprentissage génératifs offrent à ce titre un fort potentiel pour corriger et préparer au mieux les données multimodales en les transformant en des représentations sémantiquement significatives et en permettant une analyse conjointe. Ceci ouvre la voie à de nouvelles perspectives de recherche et de développement pour améliorer la qualité des données.

Pour en savoir plus

Sources bibliographiques

[1] BARBER (R.F.), CANDES (E. J.), RAMDAS (A.), and TIBSHIRANI (R.) - Predictive inference with the Jackknife+. Ann. Statist., 49(1):486–507, February, (2021).

[2] BARNETT (V.), LEWIS (T.) - Outliers in Statistical Data. - John Wiley and Sons (1994).

[3] BELKIN (M.), HSU (D. J.), and MITRA (P.) - Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate," in NeurIPS, 2018, pp. 2306–2317.

[4] BERTI-EQUILLE (L.). Learn2Clean: Optimizing the Sequence of Tasks for Web Data Preparation. Proceedings of the Web Conference 2019, pp. 2580-2586, San Francisco, May (2019).

[5] BERTOSSI (L.) - Database Repairing and Consistent Query Answering. Morgan & Claypool Publishers, (2011).

[6] BIGGIO (B.), Roli (F.) - Wild patterns: Ten years after the rise of adversarial machine learning. Pattern Recognition, Vol. 84, pp. 317-331, (2018).

[7] BISHOP (C.M.) - Training with noise is equivalent to Tikhonov regularization, Neural computation, vol. 7, no. 1, pp. 108–116, 1995.

[8] BOHANNON (P.), FAN (W.), FLASTER (M.), and RASTOGI (R.) - A cost-based model and effective heuristic for repairing constraints by value modification. ACM, pp.143-154, (2005).

[9] CHEN (Y.), ZHOU (X.S.), and HUANG (T. S.) - One-class SVM for Learning in Image Retrieval. In Proceedings 2001 IEEE International Conference on Image Processing, Vol. 1, pp.34-37, (2001).

[10] CHIANG (F.), Miller (R.) - Discovering Data Quality Rules. Proceedings of the VLDB Endowment, Vol. 1, Issue 1, August 2008, pp. 1166–1177. https://doi.org/10.14778/1453856.1453980

[11] CHRISTEN (P.) - Data Scrubbing. Chapter in the Encyclopedia of Database Systems, Springer, April 2017.

[12] CHRISTEN (P.), CHURCHES (T.) - Febrl – Freely extensible biomedical record linkage (Manual, Release 0.3), http://users.cecs.anu.edu.au/~Peter.Christen/Febrl/febrl-0.3/febrldoc-0.3/manual.html

[13] CONG (G.), FAN (W.), GEERTS (F.), JIA (X.), and MA (S.) - Improving data quality: Consistency and accuracy. In Proceedings of the Very Large Databases Conference (VLDB'07), September 23-28, 2007, Vienna, Austria. pp. 315-326, (2007).

[14] EDUARDO (S.), NAZABAL (A.), WILLIAMS (C.), and SUTTON (C.) - Robust Variational Autoencoders for Outlier Detection in Mixed-Type Data. In Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics, (2020).

[15] FAN (W.), GEERTS (F.) - Foundations of Data Quality Management. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, (2012).

[16] FEURER (M.), KLEIN (A.), EGGENSPERGER (K.), SPRINGENBERG (J. T.), BLUM (M.), and HUTTER (F.) - Efficient and robust automated machine learning, in In Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS), pp. 2962–2970, (2015).

[17] FRENAY (B.) and VERLEYSEN (M.) - Classification in the Presence of Label Noise: a Survey. IEEE Transactions on Neural Networks and Learning Systems, Vol. 25, no. 5, pp. 845-869, (2013).

[18] GANTI (V.), DAS SARMA (A.) - Data Cleaning: A Practical Perspective. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, September 2013.

[19] GETOOR (L.) and MACHANAVAJJHALA (A.) - Entity resolution: theory, practice & open challenges. PVLDB, 5(12):2018–2019, (2012).

[20] GROSSE (K.), MANOHARAN (P.), PAPERNOT (N.), BACKES (M.), and McDANIEL (P.) - On the (statistical) detection of adversarial examples. CoRR. arXiv:1702.06280 (2017). https://arxiv.org/abs/1702.06280

[21] HE (Y.), JIN (Z.) and CHAUDHURI (S.) - Auto-Transform: Learning-to-Transform by Patterns. In Proceedings of VLDB Endowment. Vol. 13, no. 11, pp. 2368-2381, (2020). http://www.vldb.org/pvldb/vol13/p2368-he.pdf

[22] HEIDARI (A.), McGRATH (J.), ILYAS (I.F. Ilyas), and REKATSINAS (T.) - HoloDetect: few-shot learning for error detection, In Proceedings of SIGMOD, 2019, pp. 829–846, (2019).

[23] KARLAS (B.), LI (P.), WU (R.), GUREL (N.M.), CHU (X.), WU (W.), and ZHANG (C.) - Nearest neighbor classifiers over incomplete information: From certain answers to certain predictions, In Proceedings of VLDB Endowment, Vol. 14, (2021).
[24] KIM (B.), XU (C.), and FOYGEL BARDER (R.) - Predictive Inference Is Free with the Jackknife+-after-Bootstrap. In Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS), (2020).
[25] KONDA (P.), DAS (S.), SUGANTHAN (P.), DOAN (A.), ARDALAN (A.), BALLARD (J.) et al. - Magellan: Toward Building Entity matching management systems. Vol. 9, no. 12, pp. 1197-1208, (2016).
[26] KRISHNAN (S.) and WU (E.) - AlphaClean: Automatic generation of data cleaning pipelines, CoRR, arXiv: 1904.11827 (2019).
[27] KRISHNAN (S.), WANG (J.), WU (M. J.), FRANKLIN (M.J.), and GOLBERG (K.). Activeclean: Interactive data cleaning for statistical modeling. In Proceedings of the VLDB Endowment, pp. 948-959 (2016).
[28] LI (M.), SOLTANOLKOTABI (M.), and OYMAK (S.) - Gradient Descent with Early Stopping is Provably Robust to Label Noise for Over-parameterized Neural Networks. Proceedings of Machine Learning Research (PMLR), Vol. 108, pp. 4313-4324. (2020). https://proceedings.mlr.press/v108/li20j.html
[29] LI (P.), X. Rao, J. Blase, Y. Zhang, X. Chu, and C. Zhang (C.) - CleanML: A study for evaluating the impact of data cleaning on ML classification tasks. In ICDE, 2021.
[30] LI (Y.), LI (J.), SUHARA (Y.), DOAN (A.), and TAN (W.-C.) - Deep Entity Matching with Pre-Trained Language Models. Proceedings of VLDB Endow. 14, 1 (Sept. 2020), 50-60, (2020). https://doi.org/10.14778/3421424.3421431
[31] LIU (F. T.), TING (K.M.), and ZHOU (Z.-H.) - Isolation forest. In 2008 Eighth IEEE International Conference on Data Mining. IEEE, pp. 413-422, (2008).
[32] MAHDAVI (M.) and ABEDJAN (Z.) - Baran: Effective error correction via a unified context representation and transfer learning," PVLDB, vol. 13, no. 11, pp. 1948–1961, 2020.
[33] MUDGAL (S.), LI (H.), REKATSINAS (T.), DOAN (A.), PARK (Y.), KRISHNAN (G.), DEEP (R.), ARCAUTE (E.), and RAGHAVENDRA (V.) - Deep Learning for Entity Matching: A Design Space Exploration. In Proceedings of the 2018 International Conference on Management of Data (Houston, TX, USA) (SIGMOD '18). New York, NY, USA, pp. 19-34, (2018). https://doi.org/10.1145/3183713.3196926
[34] NATARAJAN (N.), DHILLON (I.S.), RAVIKUMAR (P. K.), and TEWARI (A.) - Learning with Noisy Labels. In Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), Vol. 26, (2013).
[35] NAUMANN (F.), HERSCHEL (M.) - An Introduction to Duplicate Detection. Synthesis Lectures on Data Management, Morgan & Claypool Publishers, (2010).
[36] NEUTATZ (F.), CHEN (B.), ABEDJAN (Z.), and WU (E.) - From cleaning before ML to cleaning for ML. IEEE Data Eng. Bull., (2021).
[37] NOTHCUTT (C.G.), ATHALYE (A.), and MUELLER (J.) - Pervasive label errors in test sets destabilize machine learning benchmarks. arXiv preprint arXiv:2103.14749 (2021).
[38] QI (Z. X.) and WANG (H. Z.) - Dirty-Data Impacts on Regression Models: An Experimental Evaluation. In Proceedings of Database Systems for Advanced Applications (DASFAA), Lecture Notes in Computer Science, Vol. 12681. Springer, (2021). https://doi.org/10.1007/978-3-030-73194-6_6
[39] QI (Z. X.), WANG (H. Z.) and WANG (A. J.) - Impacts of Dirty Data on Classification and Clustering Models: An Experimental Evaluation. J. Comput. Sci. Technol. 36, 806–821 (2021). https://doi.org/10.1007/s11390-021-1344-6
[40] REED (S.E.), LEE (H.), ANGUELOV (D.), SZEGEDY (C.), ERHAN (D.), and RABINOVICH (A.). - Training Deep Neural Networks on Noisy Labels with Bootstrapping. In ICLR 2015, (2015). http://arxiv.org/abs/1412.6596
[41] REKATSINAS (T.), CHU (X.), ILYAS (I.F.), and RE (C.) - HoloClean: Holistic Data Repairs with Probabilistic Inference. Proceedings VLDB Endow. 10, 11, pp. 1190-1201, (Aug. 2017). https://doi.org/10.14778/3137628.3137631

[42] SARAWAGI (S.) and BHAMIDIPATY (A.) - Interactive deduplication using active learning. Proceedings of the 8th ACM SIGKDD international Conference on Knowledge discovery and data mining, July, pp. 269–278, (2002). https://doi.org/10.1145/775047.775087
[43] SCHELTER (S.), BIESSMANN (F.), LANGE (D.), RUKAT (T.), SCHMIDT (P.), SEUFERT (S.), BRUNELLE (P.), and TAPTUNOV (A.) - Unit Testing Data with Deequ. In Proceedings of the 2019 International Conference on Management of Data (SIGMOD), New York, NY, USA, pp. 1993-1996, (2019). https://doi.org/10.1145/3299869.3320210
[44] SIKDER (M.) and BATARSEH (F.A.) - Outlier Detection using AI: A . Chapter 7 in: AI Assurance, by Elsevier Academic Press. Edited by: Feras A. Batarseh and Laura Freeman Publication (2022). https://arxiv.org/abs/2112.00588
[45] STEINHARDT (J.) - Robust learning: Information theory and algorithms. Stanford University, 2018.
[46] Van DYK (D. A.) and MENG (X.-L.) - The art of data augmentation. Journal of Computational and Graphical Statistics, vol. 10, no. 1, pp. 1–50, (2001). https://doi.org/10.1016/j.patcog.2018.07.023
[47] VISENGERIYEVA (L.) and ABEDJAN (Z.) - Metadata-driven error detection, in SSDBM, pp. 1:1–1:12, (2018).
[48] WANG (J.), KRISHNAN (S.), FRANKLIN (M.J.), GOLBERG (K.), KRASKA (T.), and MILO (T.). A sample-and-clean framework for fast and accurate query processing on dirty data. In ACM SIGMOD Conference, (2014).
[49] WU (R.), CHABA (S.), SAWLANI (S.), CHU (X.), and THIRUMURUGANATHAN (S.). - ZeroER: Entity Resolution using Zero Labeled Examples. In Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference, pp. 1149-1164, (2020). https://doi.org/10.1145/3318464.3389743
[50] WU (W.), FLOKAS (L.), Wu (E.), and Wang (J.) - Complaint-driven training data debugging for query 2.0, In Proceedings of the 2019 International Conference on Management of Data (SIGMOD), 2020, pp. 1317–1334.
[51] YAKOUT (M.), BERTI-ÉQUILLE (L.), ELMAGARMID (A.) - Don't be SCAREd: use SCalable Automatic REpairing with maximal likelihood and bounded changes. In ACM SIGMOD Conference, p. 553-564, (2013).
[52] ZHANG (A.), SONG (S.), WANG (J.) and S. YU (P. S.) Time Series Data Cleaning: From Anomaly Detection to Anomaly Repairing. Proceedings VLDB Endow. 10, 10, 1046-1057, June, (2017). https://doi.org/10.14778/3115404.3115410

À lire également dans nos bases

H3700Qualité des données

H3800 Bases de données - Introduction

H3860 Bases de données relationnelles

H3248 Conception de bases de données : une méthode orientée objet et événement

H3268 Conception de bases de données - Aspects temporels

H3740 Systèmes à bases de connaissances

H3744 Extraction de connaissances à partir de données (ECD)

H3870 Entrepôts de données

H3128 Langages de bases de données : SQL et les évolutions vers l'objet

H2918 Architecture des systèmes de gestion de bases de données

H2068 Serveurs multiprocesseurs et SGBD parallélisés

H3865 Architecture client-serveur : modes d'accès aux bases de données

Événements

Conférences internationales:

- Very Large Databases (VLDB) Conference : http://vldb.org/conference.html
- ACM SIGMOD (Special Interest Group on Management of Data): https://dl.acm.org/event.cfm?id=RE227
- ACM KDD : Knowledge Discovery and Data Mining Conferences: https://www.kdd.org/conferences/
- NeurIPS : Neural Information Processing Systems : https://nips.cc/
- ICLR : International Conference on Learning Representations https://iclr.cc/

Normes et standards

- ISO/TS 8000-1:2011, Data quality — Part 1: Overview https://www.iso.org/standard/50798.html
- ISO 8000-2:2017, Data quality — Part 2: Vocabulary https://www.iso.org/standard/73456.html
- ISO 8000-8:2015, Data quality — Part 8: Information and data quality: Concepts and measuring https://www.iso.org/standard/60805.html
- ISO 8000-61:2016, Data quality — Part 61: Data quality management: Process reference model https://www.iso.org/standard/63086.html
- ISO 8000-100:2016, Data quality — Part 100: Master data: Exchange of characteristic data: Overview https://www.iso.org/standard/62392.html
- ISO 8000-110:2009, Data quality — Part 110: Master data: Exchange of characteristic data: Syntax, semantic encoding, and conformance to data specification https://www.iso.org/standard/51653.html
- ISO 8000-115:2017, Data quality — Part 115: Master data: Exchange of quality identifiers: Syntactic, semantic and resolution requirements https://www.iso.org/standard/70095.html
- ISO 8000-120:2016, Data quality — Part 120: Master data: Exchange of characteristic data: Provenance https://www.iso.org/standard/62393.html
- ISO 8000-130:2016, Data quality — Part 130: Master data: Exchange of characteristic data: Accuracy https://www.iso.org/standard/62394.html
- ISO 8000-140:2016, Data quality — Part 140: Master data: Exchange of characteristic data: Completeness https://www.iso.org/standard/62395.html
- ISO/TS 8000-150:2011, Data quality — Part 150: Master data: Quality management framework https://www.iso.org/standard/54579.html
- ISO/TS 8000-311:2012, Data quality — Part 311: Guidance for the application of product data quality for shape (PDQ-S) https://www.iso.org/standard/60010.html