

Construction de variables à l'aide de classifieurs comme aide à la régression : une évaluation empirique

Colin Troisemaine*, Vincent Lemaire*

*Orange Labs Lannion,

Résumé. Cet article propose une méthode de création automatique de variables (pour la régression) qui viennent compléter les informations contenues dans le vecteur initial d'entrée, les variables explicatives. Notre méthode fonctionne comme une étape de prétraitement dans laquelle les valeurs continues de la variable à régresser sont discrétisées en un ensemble d'intervalles. Ces intervalles permettent de définir des seuils de valeurs. Ensuite, des classifieurs sont entraînés pour prédire si la valeur à régresser est inférieure ou égale à chacun de ces seuils. Les différentes sorties des classifieurs sont ensuite concaténées sous la forme d'un vecteur additionnel de variables qui vient enrichir le vecteur initial de variables explicatives natives du problème de régression. Le système implémenté peut donc être considéré comme un outil de prétraitement générique. Nous avons testé la méthode d'enrichissement proposée avec 5 types de régresseurs et l'avons évalué dans 33 jeux de données de régression. Nos résultats expérimentaux confirment l'intérêt de l'approche.

1 Introduction

Les techniques d'apprentissage peuvent se découper en deux grandes familles selon leur vocation principale : celles servant à décrire les données (méthodes descriptives) et celles permettant de prédire un phénomène (plus ou moins) observable (méthodes prédictives). Les méthodes prédictives permettent de prévoir et d'expliquer à partir d'un ensemble de données étiquetées un ou plusieurs phénomènes (plus ou moins) observables. Dans le cas de la régression il s'agira de prévoir la valeur d'une variable numérique (noté y), par exemple le montant d'une facture, à l'aide d'un ensemble de variables explicatives (un vecteur noté X).

Dans le cas de l'apprentissage automatique, on cherchera à apprendre une fonction f telle que $y = f(X)$ à l'aide d'un algorithme d'apprentissage automatique et d'un ensemble d'apprentissage, un ensemble de N couples entrée-sortie (X_i, y_i) , $i = 1, \dots, N$. Lors de cette étape de modélisation, il existe souvent le besoin de créer de nouvelles variables qui décrivent mieux le problème et permettent au modèle d'atteindre de meilleures performances. C'est ce qu'on appelle le "processus d'ingénierie de création de nouvelles variables explicatives" (Sondhi, 2009). Dans ce cas, on espère que les nouvelles variables (un vecteur qui sera ici noté X') apporteront une information additionnelle. L'automatisation de la génération de ces "nouvelles variables" permet d'extraire des informations plus utiles et significatives des données, dans un cadre qui peut être appliqué à n'importe quel problème. Ce qui permet à l'ingénieur en apprentissage automatique de consacrer plus de temps à des tâches plus utiles.

L'objectif de l'article ici présenté est de proposer une méthode de création automatique de variables (dans le cas de la régression) qui viennent compléter les informations contenues dans le vecteur X pour prédire les valeurs de la variable dépendante y . La méthode proposée transforme tout d'abord le problème de régression en plusieurs sous-problèmes de classification, puis intègre les résultats sous forme de variables additives (X'). Le vecteur 'augmenté', $X'' = X \cup X'$, est ensuite mis en entrée de régresseurs usuels afin de mesurer l'apport des variables créées. L'intérêt de l'approche est présenté sous la forme d'une étude expérimentale détaillée.

2 Proposition

La méthode proposée repose sur une transformation du problème de régression en un problème de classification pour construire les variables additionnelles. Cette idée a déjà été suggérée, mais notre approche diffère de l'état de l'art comme expliqué ci-dessous dans la section 2.1. Cette différence ayant été présentée, la section 2.2 décrit en détail la méthode proposée et la section 2.3 les choix qui ont été faits pour son implémentation.

2.1 Travaux liés

Résoudre un problème de régression en s'appuyant sur des modèles de classification est une approche qui a déjà été explorée. Ce processus a été décrit dans de nombreux articles (Ahmad et al., 2012, 2018; Janssen et Fürnkranz, 2010, 2011; Memon et al., 2019; Torgo et Gama, 1997) et se compose généralement de deux étapes principales : (i) la discrétisation (également appelée partitionnement ou binning) de la variable cible afin de permettre l'utilisation de classifieurs sur le jeu de données ; (ii) la prédiction de la régression est alors généralement réalisée en calculant la moyenne ((Ahmad et al., 2012, 2018; Janssen et Fürnkranz, 2010; Memon et al., 2019)) ou la médiane ((Ahmad et al., 2018; Janssen et Fürnkranz, 2011; Torgo et Gama, 1997)) des instances à l'intérieur du fragment de la sortie discrétisée que le classificateur a prédit.

La méthode que nous proposons et présentons ci-après diffère de ces travaux car les classifieurs utilisés par la méthode ont uniquement pour objectif d'ajouter des variables explicatives complémentaires aux variables explicatives initiales (variables natives). Le vecteur augmenté est ensuite positionné en entrée d'un régresseur usuel. Ce régresseur prédit directement la variable cible sans nouvelle opération de transformation ou d'estimation. Comme le montrera la section suivante, la méthode proposée est liée à une estimation conditionnelle de la fonction de densité de y . Il serait alors intéressant de tester d'autres méthodes de création de variables dans ce même cadre mais moins gourmandes en calcul (Rothfuss et al., 2019; Holmes et al., 2007; Tutz, 2021; Langford et al., 2012) dans de futurs travaux. Le principe général demeurant néanmoins le même.

2.2 Principe général

L'idée est de tirer profit de manière opportuniste des progrès réalisés ces dernières années par les classifieurs de la littérature. Le principe de la première étape de la méthode proposée est de transformer le problème de régression en un (ou plusieurs) problème(s) de classification

en *discréti*sant l'espace de variation de la variable à régresser. Cette étape consiste à définir des seuils (S) sur l'espace de la variable à régresser (voir figure 1).

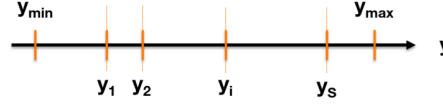


FIG. 1 – Exemple de seuils

Ces seuils seront définis à l'aide des valeurs de la variable à régresser de l'ensemble d'entraînement. Ils permettront de définir des classes d'appartenance ($C = \{C_1, \dots, C_i, \dots, C_S\}$). On pourra citer comme exemple des classes définies sur des seuils d'infériorité de valeur ($C_i := \mathbb{K}_{y \leq y_i}$) ou encore des classes définies sur des appartenances à des intervalles de valeurs ($C_i := \mathbb{K}_{y \in [y_i, y_{i+1}]}$), etc.

Une fois les classes encodées et un (ou plusieurs) classifieurs entraînés (à l'aide de l'ensemble d'entraînement) il est alors possible de prédire l'appartenance des individus aux classes préalablement définies. Les prédictions du ou des classifieurs sur chacun des individus seront alors utilisées pour créer un nouvel ensemble de données "étendu", que ce soit pour l'ensemble d'entraînement ou pour l'ensemble de test.

La figure 2 illustre l'extension du vecteur X (ayant d composantes) au vecteur X'' (ayant $d + S$ composantes). La section 2.3 décrira plus en détail de quoi est composé le vecteur $X' = \{X'_1, \dots, X'_i, \dots, X'_S\}$ provenant de la prédiction du ou des classifieurs.

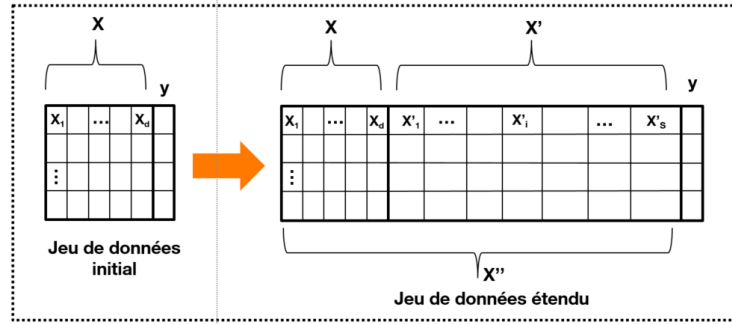


FIG. 2 – Extension de l'ensemble de données.

Une fois le vecteur X ayant été "étendu", il est possible de réaliser l'entraînement et la prédiction du modèle de régression sur ces nouvelles données. On pourra alors comparer la performance du régresseur lorsqu'on l'entraîne sur l'ensemble de données original (muni uniquement de X) à sa performance lorsqu'on utilise l'ensemble étendu ($X'' = X \cup X'$). L'hypothèse étant que le modèle de régression aura une meilleure intuition de la position générale des individus dans l'espace de la variable à régresser et présentera de meilleurs résultats.

La figure 3 résume le processus suivi par la méthode proposée : (i) la première étape consiste à transformer le problème de régression en problème de classification. Pour ce faire,

Construction de variables à l'aide de classifieurs comme aide à la régression

la variable cible y est d'abord discrétisée (voir section 2.3), puis les classes sont définies à l'aide des seuils ainsi définis. (ii) Dans une deuxième étape, les classifieurs sont entraînés en utilisant les variables descriptives initiales de X et les nouvelles classes dérivées de y . Puis la prédiction, des classifieurs utilisant le vecteur X initial, est utilisée pour extraire de nouvelles caractéristiques, c'est-à-dire X' . (iii) Enfin, le modèle de régression peut être entraîné à l'aide du vecteur $X'' = X \cup X'$. La méthode proposée repose donc sur un mécanisme de discrétisation et d'encodage de classes. Ces deux processus peuvent être mis en place de différentes façons. Nous proposons un exemple d'implémentation ci-après dans la section 2.3.

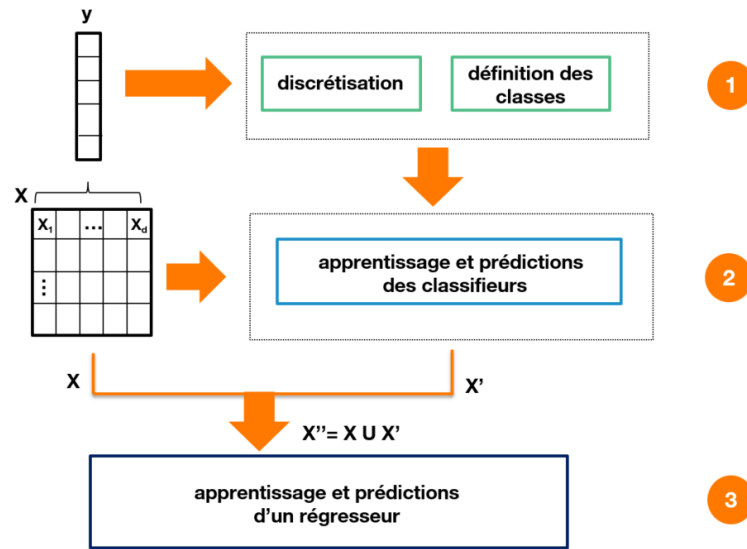


FIG. 3 – Schématisation du principe général de la méthode.

2.3 Implémentation

L'un des aspects les plus importants de la méthode proposée est l'association de classes aux instances. Ces classes seront utilisées lors de l'apprentissage des classifieurs. Ce processus repose sur deux étapes : la définition de seuils et l'encodage des classes. Pour la définition du *placement des seuils*, ce qui revient à une discrétisation non-supervisée de la variable à régresser, il existe dans la littérature de nombreuses possibilités telles que "EqualWidth", "EqualFreq", etc. Pour l'*encodage des classes*, il est possible de poser le problème soit comme un problème de classification à S classes (où S désigne le nombre de seuils), soit comme S problèmes de classification binaire, soit comme un problème de classification multi-labels.

Les choix de la méthode de discrétisation et du nombre de seuils sont liés et ce choix représente un compromis entre (i) l'efficacité en classification et (ii) l'apport d'information pour le problème initial de régression. Dans le cas d'un unique classifieur qui prédirait l'intervalle d'appartenance des individus, il semble évident que plus S sera grand, plus l'information apportée au régresseur sera précise, mais plus le problème deviendra difficile à apprendre. En

revanche, si on pose le problème comme S problèmes de classification binaire, chaque classifieur aura un problème de même difficulté à résoudre, indépendamment de la valeur de S . Des tests préliminaires effectués sur l'ensemble des jeux de données décrits Section 3.1 ont permis de vérifier ce comportement. Le choix final pour les différents éléments associés à la méthode proposée dans cet article sont décrits ci-dessous.

Pour la **méthode de discrétisation** - La méthode "Equal Frequency" a été choisie car contrairement à "Equal Width", elle ne risque pas de créer d'intervalles qui ne contiennent aucun individu. Elle permet également de s'assurer de ne pas poser problème de classification où la classe minoritaire (seuil extrême à gauche, voir Figure 1) représente moins de $\frac{1}{S+1}$ pourcent des individus.

Pour la nature du **problème de classification** - Lors des tests préliminaires, le fait de poser le problème en S problèmes de classification binaire a donné de meilleures performances, tant en apprentissage qu'en déploiement. Les résultats ont aussi été plus robustes (meilleure généralisation), ce qui est un point important si l'on veut que les variables ajoutées soient bénéfiques au régresseur. Les classifieurs utilisés dans le reste de cet article sont les Forêts Aléatoires (Breiman, 2001) de Scikit-Learn (Pedregosa et al., 2011) avec 100 arbres (mais tout autre classifieur performant et robuste pourrait être utilisé) avec leurs paramètres par défaut.

Pour le **nombre de seuils** - La première intuition (qui a été confirmée lors de l'expérimentation) est que plus le nombre de seuils défini est grand, plus le gain de performance est potentiellement important. En effet, plus on définit de seuils, plus la discrétisation est fine et plus la prédiction des classifieurs s'approchera de la valeur réelle de régression. Lors la phase expérimentale de cet article, nous illustrerons la performance de notre méthode en fonction du nombre de seuils défini. Le nombre de seuils défini sur l'espace de la variable à régresser est donc un hyper-paramètre de notre méthode.

Pour la définition des **classes associées aux seuils** - Le choix retenu s'est porté sur des classes liées à des seuils d'infériorité de valeur ($C_i := \mathbb{1}_{y \leq y_i}$) (voir Section 2.2 et Figure 1).

Pour les **variables extraites** des classifieurs - Chaque classifieur sera entraîné à prédire si les données que l'on lui fournit sont inférieures ou supérieures au seuil auquel une classe lui est associée. La méthode se voulant générique, il a été décidé d'extraire les probabilités conditionnelles prédites par chaque classifieur. En effet, cette information est accessible pour une large majorité des classifieurs de la littérature. Les probabilités conditionnelles¹ de la classe 1 (c'est à dire si $y \leq y_i$) prédites par chaque classifieur composeront donc le vecteur $X' = \{X'_1, \dots, X'_i, \dots, X'_S\} = \{P(C_1 = 1|X), \dots, P(C_i = 1|X), \dots, P(C_S = 1|X)\}$; X' qui sera donc ajouté au vecteur de données initial (X).

3 Protocole experimental

3.1 Jeux de données et prétraitements

Jeux de données : Pour réaliser l'analyse de la méthode proposée, nous avons sélectionné une large collection de jeux de données de régression provenant du UCI Repository (Dua et Graff, 2017) ou de Kaggle (kag). Au total, 33 jeux de données ont été utilisés, parmi lesquels 23 sont constitués de plus de 10 000 individus. Les 10 jeux de données restants allant de 1 030

1. Note : les classifieurs retenus étant des forêt aléatoires, la probabilité conditionnelle correspondra à la proportion d'arbres ayant voté pour la classe concernée.

à 9 568 individus (voir Table 1). Cette sélection a été largement influencée par l'article de M. Fernandez-Delgado et al. (Fernández-Delgado et al., 2019). Les jeux de données choisis sont d'après la catégorisation de Fernandez-Delgado et al. "*grands et difficiles*". Certains de ces jeux de données comprenaient plusieurs sous-ensembles de données avec différentes variables cibles à régresser, et donc différents problèmes de régression. Il a été choisi de ne pas inclure plusieurs problèmes de régression provenant des mêmes ensembles de données, afin de pas donner plus de poids à certains ensembles de données et créer un biais dans la comparaison des méthodes qui pourrait favoriser une méthode par rapport à une autre. Le lecteur pourra se référer à la colonne "nom de l'ensemble de données" de la Table 1 qui désigne le problème de régression utilisé pour chaque jeu de données.

Prétraitements : A l'identique de (Fernández-Delgado et al., 2019), deux prétraitements ont été réalisés avant d'entrer dans le processus décrit dans la Figure 3 (voir Section 2.2) : (i) le recodage des variables catégorielles à l'aide d'un codage disjonctif complet ; (ii) la suppression des variables de dates, des variables constantes, des identifiants d'individus, des variables colinéaires et des autres variables potentiellement à 'régresser'. Trois autres prétraitements ont été réalisés sur chaque "fold" d'apprentissage. Ces derniers sont dans l'ordre de réalisation : (i) normalisation des variables numériques (centrage - réduction) ; (ii) normalisation de la variable à régresser (centrage - réduction) puis transformation à l'aide de Box-Cox (Atkinson, 2020) ; (iii) création des seuils pour la définition des classes tel que décrit précédemment. Pour chacun de ces trois prétraitements, les statistiques qui leurs sont associées ont été calculées sur l'ensemble d'entraînement puis appliquées sur les ensembles d'entraînement et de test. Dans la présentation ci-après des résultats, les résultats en RMSE (voir la métrique utilisée ci-dessous) seront donnés sans effectuer la transformation inverse de la fonction Box-Cox. Enfin les exemples à valeurs manquantes ont été retirés du jeu de données initial. Des statistiques des jeux de données après prétraitements sont présentés dans la Table 1.

Découpage Train-Test et optimisation des modèles : Chacun des jeux de données a été découpé en un "10-fold cross validation" permettant d'avoir 10 jeux d'entraînement et de test. Pour les trois modèles nécessitant une optimisation de leurs paramètres (AR, FA et XGB voir Section 3.3) : 30% de l'ensemble d'entraînement a été réservé pour optimiser les paramètres du modèle dans un processus de "grid-search", permettant ainsi souvent d'éviter un sur-apprentissage. Puis, muni des "bons" paramètres d'apprentissage, le modèle a été entraîné à l'aide des 70% restant de l'ensemble d'entraînement. Enfin, une fois le modèle appris pour ce fold, ses performances sont mesurées. Ce processus rigoureux a été réalisé pour tous les jeux de données, tous les folds et toutes les valeurs de S , résultant ainsi en l'apprentissage de milliers de modèles, mais permettant un test rigoureux de la méthode proposée.

3.2 Métrique utilisée pour les résultats

Lorsque l'on compare des modèles de régression l'objectif est de trouver un modèle sans sur ou sous-adaptation qui atteigne une faible erreur de généralisation, ce qui caractérise sa performance de prédiction. La littérature propose différentes métriques comme la racine carrée de la moyenne des erreurs quadratique (acronyme RMSE en anglais), la moyenne des erreurs en valeur absolue (MAE), le R-Square, etc ... Il ne semble pas avoir de consensus sur la "meilleure" métrique à utiliser même si certains articles proposent des comparaisons (Botchkarev, 2019). Dans la suite des résultats, la RMSE a été choisie et est définie telle que :

| Jeu de données | #individus | # variables | target | Jeu de données | #individus | # variables | target |
|----------------------|------------|-------------|----------------------|---------------------|------------|-------------|-------------------|
| 3Droad | 434,874 | 3 | altitude | geo-lat | 1,059 | 116 | latitude |
| air-quality-CO | 1,230 | 8 | PT08.S1(CO) | greenhouse-net | 955,167 | 15 | synthetic var |
| airfoil | 1,503 | 5 | scaled sound | KEGG-reaction | 65,554 | 27 | edge count |
| appliances-energy | 19,735 | 26 | appliances | KEGG-relation | 54,413 | 22 | clustering coef |
| beijing-pm25 | 41,758 | 14 | PM2.5 | online-news | 39,644 | 59 | shares |
| temp-forecast-bias | 7,752 | 22 | Next_Tmax | video-transcode | 68,784 | 26 | utime |
| bike-hour | 17,379 | 17 | count | pm25-chengdu-us | 27,368 | 20 | PM_US Post |
| blog-feedback | 60,021 | 18 | target | park-total-UPDRS | 5,875 | 16 | total_UPDRS |
| buzz-twitter | 583,250 | 70 | discussions | physico-protein | 45,730 | 9 | RMSD |
| combined-cycle | 9,568 | 4 | PE | production-quality | 29,184 | 17 | quality |
| com-crime | 1,994 | 122 | Violent crimes | CT-slices | 53,500 | 384 | reference |
| com-crime-unnorm | 2,215 | 134 | Violent crimes | gpu-kernel-perf | 241,600 | 14 | Run1 (ms) |
| compress-stren | 1,030 | 8 | compressive strength | SML2010 | 4,137 | 26 | indoor temp |
| cond-turbine | 11,934 | 15 | gt turbine | seoul-bike-sharing | 8,760 | 9 | Rented Bike Count |
| cuff-less | 61,000 | 2 | APB | uber-location-price | 205,670 | 5 | fare amount |
| electrical-grid-stab | 10,000 | 12 | stab | year-prediction | 515,345 | 90 | Year |
| facebook-comment | 40,949 | 53 | target | | | | |

TAB. 1 – Description des 33 jeux de données. La première colonne désigne le nom du jeu (ou du sous-jeu) de données provenant de l'UCI ou de Kaggle. La deuxième et la troisième colonne donnent respectivement le nombre d'individus et le nombre de variables (explicatives) initial du jeu de données (après les étapes de prétraitements). Enfin la quatrième colonne indique le nom de la variable cible à régresser.

$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$, où n est le nombre d'individus, y_i la valeur désirée et \hat{y}_i la sortie d'un régresseur.

3.3 Régresseurs testés

Lors des expérimentations, cinq régresseurs utilisant des cadres différents ont été utilisés. Ils sont décrits brièvement ci-dessous, tout en étant largement connus par la communauté de l'apprentissage automatique.

- Régression Linéaire (LR²) : La régression désigne le processus d'estimation d'une variable numérique continue à l'aide d'autres variables qui lui sont corrélées. Cela signifie que les modèles de régression prennent la forme de $y = f(X)$ où y peut prendre un ensemble infini de valeurs. Dans cet article, nous avons choisis une régression linéaire multivariée dont le modèle, f , est déterminé par la méthode des moindres carrés. Nous avons utilisé la version 0.24.2 de Scikit-Learn. Ce modèle ne nécessite pas d'optimisation de ses paramètres.

- Arbre de Régression (DT) : Les arbres de décision permettent de prédire une quantité réelle qui, dans le cas de régression, est une valeur numérique. Usuellement, les algorithmes pour construire les arbres de décision sont construits en divisant l'arbre du sommet vers les feuilles en choisissant à chaque étape une variable d'entrée qui réalise le meilleur partage de l'ensemble d'objets. Dans le cas des arbres de régression, on cherche à maximiser la variance inter-classes (avoir des sous-ensembles dont les valeurs de la variable-cible sont le plus dispersées possibles). Dans cet article, nous avons utilisé la version 0.24.2 de Scikit-Learn. Ce modèle nécessite une optimisation de ses paramètres.

2. Dans la suite les acronymes anglais seront utilisés.

- **Forêt aléatoire (RF)** : Les forêts d'arbres décisionnels (ou forêts aléatoires de l'anglais Random Forest (RF) régresseur) ont été proposées par Ho en 1995 (Ho, 1995) puis étendu par Leo Breiman et Adele Cutler (Breiman, 2001). Cet algorithme combine les concepts de sous-espaces aléatoires et de bagging. L'algorithme des forêts d'arbres décisionnels effectue un apprentissage sur de multiples arbres de décision entraînés sur des sous-ensembles de données légèrement différents. Dans cet article, nous avons utilisé la version 0.24.2 de Scikit-Learn. Ce modèle nécessite une optimisation de ses paramètres.

- **XGBoost (XGB)** : XGBoost (Chen et Guestrin, 2016) est une méthode de boosting. Elle combine séquentiellement des apprenants faibles qui présenteraient individuellement de mauvaises performances pour améliorer la prédiction de l'algorithme complet. Un apprenant faible est un régresseur qui a une faible performance de régression. Dans cet algorithme de boosting, des poids élevés seront associés aux apprenants faibles ayant une bonne précision, et à l'inverse, des poids plus faibles aux apprenants faibles ayant une mauvaise précision. Dans la phase d'entraînement, des poids élevés sont associés aux données qui ont été mal "apprises" afin que l'apprenant faible suivant dans la séquence se concentre davantage sur ces données. Dans cet article, nous avons utilisé la version 1.4.1 de XGBoost. Ce modèle nécessite une optimisation de ses paramètres.

- **Naïve Bayes Pondéré (SNB)** : Dans le contexte de la régression, un régresseur naïf de Bayes (NB) dont les variables sont pondérées par des poids (NBP) peut être obtenu à l'aide de deux étapes. Dans un premier temps, pour chaque variable explicative, on crée une grille 2D pour estimer $P(X, y)$ (voir par exemple (Boullé, 2006)). Ensuite, dans une deuxième étape, toutes les variables sont regroupées dans un algorithme Forward Backward (Boullé, 2007) pour estimer leur informativité dans le contexte d'un régresseur Naïf de Bayes. A la fin de la deuxième étape, le modèle final (à déployer) est un Naïve Bayes (qui utilise la discrétisation 2D trouvée dans la première étape) où les variables sont pondérées (les poids sont trouvés dans la deuxième étape). Dans cet article nous avons utilisé le Naïve Bayes Pondéré produit par le logiciel Khiops (www.khiops.com). Nous avons demandé et obtenu gratuitement une licence provisoire du logiciel ainsi que sa version pyKhiops utilisable dans Scikit-Learn. Ce modèle ne nécessite pas d'optimisation de ses paramètres.

4 Résultats

- **Résultats illustratifs** - La Figure 4 illustre le comportement de la méthode proposée : A gauche pour la régression linéaire et le jeu de données "KEGG Metabolic Reaction", au milieu et à droite pour le jeu de données "SML 2010" respectivement pour le régresseur naïf de Bayes et la forêt aléatoire. Ces 3 figures sont assez représentatives des résultats obtenus, avec une décroissance plus ou moins marquée de la RMSE versus la valeur de S et le régresseur considéré. Cette décroissance est plus marquée en début de courbe et plafonne ensuite pour un gain moins marqué au-delà de 16 seuils. La section suivante donnera les résultats obtenus pour chacun des jeux de données.

- **Tables de résultats sur l'ensemble des jeux de données** - Les résultats obtenus sont présentés en détail dans la Table 2 pour un nombre de seuils égal à 32 ($S = 32$). Pour chaque jeu de données, cette table donne les résultats en test (résultats moyen sur les 10 "fold" de test (voir Section 3.1)) pour chacun des cinq régresseurs pour lesquels l'ajout du vecteur X' a été testé. Dans cette table, une valeur en gras indique une différence significative entre les résultats

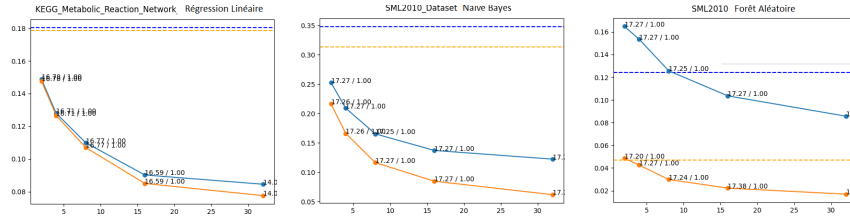


FIG. 4 – S (axe horizontal) versus RMSE (axe vertical). Les lignes en pointillés représentent la performance initiale du régresseur en bleu pour l'ensemble de test et en orange pour l'ensemble d'entraînement. Les courbes en trait plein représentent la performance du régresseur avec la méthode proposée, en reprenant le même code couleur.

avec le vecteur X "Natif" et le vecteur X' "Aug" (Augmenté) selon test de Student apparié (p-value à 5%). La dernière ligne du tableau indique le nombre de défaites, égalités et victoires de "Aug" vis-à-vis de "Natif". On s'aperçoit que l'ajout du vecteur X' profite essentiellement à 3 des régresseurs : La régression linéaire, l'arbre de régression et le régresseur naïf de Bayes. Pour la forêt aléatoire et XGboost, les gains et/ou pertes sont assez limités.

L'avant dernière ligne du tableau présente la RMSE moyenne obtenue sur l'ensemble des jeux de données (purements à titre indicatif) confirmant l'intérêt de la méthode proposée pour trois des cinq régresseurs. Cette moyenne doit être regardée avec précaution. Comme chaque ensemble de données est un problème d'une difficulté unique, l'échelle des erreurs diffère entre chaque ensemble de données. Ainsi, si la nouvelle RMSE *moyenne* en test est plus élevée, cela ne signifie pas nécessairement que les régresseurs ont de moins bonnes performances en moyenne. C'est pourquoi nous présentons ci-après dans la section suivante une autre vue sur les résultats afin de poursuivre leur analyse.

• **Diagramme Critique** - Nous présentons Figure 5 des diagrammes critiques comparant entre eux les résultats obtenus par chaque régresseur par le biais du test post-hoc de Nemenyi, effectué après un test des rangs signés de Friedman des RMSE, cela en considérant tous les jeux de données.

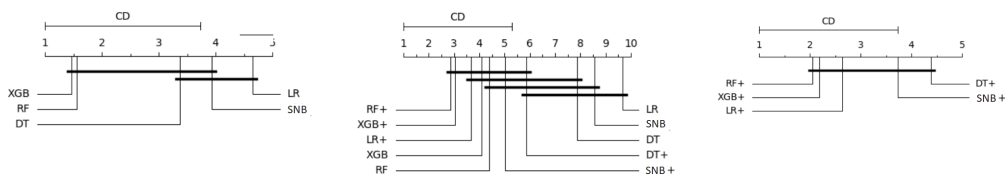


FIG. 5 – Diagramme Critique : À gauche les régresseurs sans la méthode proposée, au centre les régresseurs avec et sans la méthode proposée et à droite les régresseurs uniquement avec la méthode proposée (indiquée par un '+')

On peut observer sur le diagramme de gauche de la figure 5 que le régresseur XGBoost présente les meilleures performances en obtenant le meilleur rang moyen de RMSE, suivi de près par la forêt aléatoire (RF). Ces deux régresseurs sont largement plus performants que les trois autres régresseurs, la régression linéaire est le modèle le moins performant. Elle est suivie

Construction de variables à l'aide de classifieurs comme aide à la régression

| | Linear Regr. | | Decision Tree | | Random Forest | | XGBoost | | SNB | |
|----------------------|--------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
| Dataset name | Natif | Aug | Natif | Aug | Natif | Aug | Natif | Aug | Natif | Aug |
| 3Droad | 0,9867 | 0,0930 | 0,1213 | 0,0831 | 0,0908 | 0,0792 | 0,0935 | 0,0781 | 0,5580 | 0,0827 |
| air-quality-CO | 0,3269 | 0,2672 | 0,3255 | 0,2727 | 0,2667 | 0,2642 | 0,2682 | 0,2633 | 0,3428 | 0,2768 |
| airfoil | 0,7176 | 0,2351 | 0,4373 | 0,3194 | 0,2987 | 0,2865 | 0,2814 | 0,2752 | 0,7709 | 0,2963 |
| appliances-energy | 0,8260 | 0,4865 | 0,7431 | 0,5267 | 0,5456 | 0,5164 | 0,5790 | 0,5203 | 0,8354 | 0,5206 |
| beijing-pm25 | 0,7833 | 0,4180 | 0,6013 | 0,3948 | 0,4168 | 0,3874 | 0,4149 | 0,3885 | 0,8082 | 0,3931 |
| temp-forecast-bias | 0,4749 | 0,2847 | 0,4668 | 0,2848 | 0,3237 | 0,2771 | 0,3085 | 0,2773 | 0,4782 | 0,2907 |
| bike-hour | 0,7163 | 0,2547 | 0,3294 | 0,2564 | 0,2499 | 0,2208 | 0,2500 | 0,4858 | 0,2534 | 0,2534 |
| blog-feedback | 0,8307 | 0,6434 | 0,6741 | 0,6694 | 0,6481 | 0,6618 | 0,6444 | 0,6652 | 0,7013 | 0,6601 |
| buzz-twitter | 0,7248 | 0,2183 | 0,2299 | 0,2265 | 0,2170 | 0,2200 | 0,2162 | 0,2205 | 0,2238 | 0,2191 |
| combined-cycle | 0,2702 | 0,1967 | 0,2647 | 0,2014 | 0,2079 | 0,1952 | 0,2041 | 0,1950 | 0,2570 | 0,1995 |
| com-crime | 0,6312 | 0,5485 | 0,6418 | 0,5552 | 0,5525 | 0,5505 | 0,5665 | 0,5503 | 0,5612 | 0,5523 |
| com-crime-unnorm | 0,6751 | 0,6031 | 0,7036 | 0,6251 | 0,6163 | 0,6157 | 0,6186 | 0,6124 | 0,6453 | 0,6340 |
| compress-stren | 0,6331 | 0,2732 | 0,4519 | 0,3173 | 0,3137 | 0,2896 | 0,2790 | 0,2955 | 0,5962 | 0,2959 |
| cond-turbine | 0,3002 | 0,1096 | 0,1764 | 0,1094 | 0,1117 | 0,1064 | 0,1062 | 0,1094 | 0,4481 | 0,1096 |
| cuff-less | 0,7996 | 0,5791 | 0,5165 | 0,6255 | 0,5025 | 0,6105 | 0,5012 | 0,6225 | 0,5450 | 0,5922 |
| electrical-grid-stab | 0,5961 | 0,3156 | 0,5545 | 0,3245 | 0,3352 | 0,3068 | 0,2634 | 0,3092 | 0,5951 | 0,3157 |
| facebook-comment | 0,6828 | 0,4242 | 0,4655 | 0,4429 | 0,4162 | 0,4384 | 0,4095 | 0,4397 | 0,5014 | 0,4347 |
| geo-lat | 0,8752 | 0,8066 | 0,9594 | 0,8538 | 0,8321 | 0,8287 | 0,8540 | 0,8353 | 0,8668 | 0,8408 |
| greenhouse-net | 0,6492 | 0,5418 | 0,5397 | 0,5686 | 0,5158 | 0,5638 | 0,5137 | 0,5618 | 0,6016 | 0,5599 |
| KEGG-reaction | 0,1803 | 0,0845 | 0,0919 | 0,0861 | 0,0837 | 0,0835 | 0,0835 | 0,0840 | 0,1172 | 0,0855 |
| KEGG-relation | 0,6342 | 0,1750 | 0,2454 | 0,1748 | 0,1782 | 0,1700 | 0,1809 | 0,1715 | 0,4314 | 0,1738 |
| online-news | 1,1822 | 1,0677 | 0,9516 | 0,9622 | 0,9177 | 0,9521 | 0,9124 | 0,9521 | 0,9228 | 0,9942 |
| video-transcode | 0,3975 | 0,0561 | 0,0846 | 0,0652 | 0,0604 | 0,0597 | 0,0590 | 0,0568 | 0,3704 | 0,0627 |
| pm25-chengdu-us-post | 0,8022 | 0,4283 | 0,5578 | 0,4164 | 0,4111 | 0,4084 | 0,4125 | 0,4079 | 0,7716 | 0,4131 |
| park-total-UPDRS | 0,9472 | 0,7816 | 0,9416 | 0,8057 | 0,8020 | 0,7893 | 0,8196 | 0,7860 | 0,9428 | 0,8117 |
| physico-protein | 0,8453 | 0,5179 | 0,7676 | 0,5338 | 0,5528 | 0,5218 | 0,5744 | 0,5249 | 0,8494 | 0,5330 |
| production-quality | 0,4872 | 0,2807 | 0,3886 | 0,2830 | 0,2836 | 0,2779 | 0,2794 | 0,2781 | 0,3790 | 0,2832 |
| CT-slices | 1,9533 | 0,0504 | 0,1332 | 0,0586 | 0,0544 | 0,0418 | 0,0693 | 0,0355 | 0,1104 | 0,0376 |
| gpu-kernel-perf | 0,6347 | 0,0696 | 0,0300 | 0,0557 | 0,0246 | 0,0501 | 0,0228 | 0,0500 | 0,5927 | 0,0620 |
| SML2010 | 0,2536 | 0,1157 | 0,2179 | 0,1036 | 0,1241 | 0,0857 | 0,1058 | 0,0920 | 0,3473 | 0,1221 |
| seoul-bike-sharing | 0,6990 | 0,4839 | 0,5801 | 0,5217 | 0,5101 | 0,5149 | 0,5171 | 0,5139 | 0,5891 | 0,5169 |
| uber-location-price | 0,9998 | 0,4589 | 0,6153 | 0,4789 | 0,4635 | 0,4699 | 0,4705 | 0,4688 | 0,8419 | 0,4879 |
| year-prediction | 0,8569 | 0,8079 | 0,8783 | 0,8137 | 0,8059 | 0,8062 | 0,7929 | 0,8131 | 0,8822 | 0,8138 |
| Moyenne | 0,7083 | 0,3842 | 0,4754 | 0,3945 | 0,3856 | 0,3842 | 0,3831 | 0,3850 | 0,5749 | 0,3917 |
| Déf / Egal / Vict | 0 / 4 / 29 | | 3 / 2 / 28 | | 8 / 16 / 9 | | 10 / 14 / 9 | | 2 / 2 / 29 | |

TAB. 2 – Résultats pour chaque jeu de données en test (résultats moyen sur les 10 "fold" de test (voir Section 3.1)) pour $S = 32$.

de près par le régresseur naïf de Bayes et l'arbre de régression. La figure 5 au centre compare les performances de tous les régresseurs, avec et sans la méthode proposée. Le premier point que l'on peut observer est que statistiquement, tous les régresseurs munis de X' ont un meilleur rang moyen que leur alter-ego muni uniquement de X , ceci est un résultat encourageant : aucun régresseur n'a été négativement affecté par la méthode proposée, puisque tous les régresseurs qui ont utilisé cette méthode sont mieux classés que leurs versions de base. Différents groupes sont à nouveau présents et le lecteur pourra comparer le changement de classement des différents régresseurs en comparant la figure de gauche à celle du milieu. Enfin, le diagramme de droite de la figure 5 montre que, lorsque la méthode proposée est utilisée, les régresseurs ne sont plus différenciables (un seul groupe comparé à la figure de gauche) selon le test de Nemenyi, même s'ils ont des rangs moyens différents.

5 Conclusion

Cet article a proposé une méthode de création automatique de variables (dans le cas de la régression) qui viennent compléter les informations contenues dans le vecteur initial d'entrée, les variables explicatives. La méthode fonctionne comme une étape de prétraitement dans laquelle les valeurs continues de la variable à régresser sont discrétisées en un ensemble d'intervalles. Ces intervalles permettent de définir des seuils de valeurs. Ensuite, des classifieurs sont entraînés pour prédire si la valeur à régresser est inférieure ou égale à chacun de ses seuils. Les différentes sorties des classifieurs sont ensuite concaténées sous la forme d'un vecteur additionnel de variables qui vient enrichir le vecteur initial de variables explicatives natives du problème de régression. Les résultats sont encourageants, même s'ils sont surtout bénéfiques à trois des cinq régresseurs pour lesquels la méthode a été testée. Une première amélioration serait d'extraire des classifieurs un vecteur plus informatif, par exemple les identifiants des feuilles des arbres. Une deuxième amélioration pourrait être de concevoir une architecture neuronale regroupant l'ensemble des étapes de la méthode proposée.

La méthode proposée ouvre aussi certaines perspectives. Le lecteur attentif aura probablement remarqué que tel que défini dans l'implémentation proposée, le vecteur X' est en fait une estimation de la densité cumulée conditionnelle de y connaissant X . Il serait donc possible via un calcul d'espérance de se passer totalement des régresseurs dans le cas où S serait suffisamment élevé pour avoir une assez bonne estimation de cette densité cumulée. Ce dernier point fera probablement l'objet de travaux futurs.

Références

- Kaggle. <https://www.kaggle.com>.
- Ahmad, A., S. M. Halawani, et I. Albidewi (2012). Novel ensemble methods for regression via classification problems. *Expert Syst. Appl.* 39, 6396–6401.
- Ahmad, A., S. S. Khan, et A. Kumar (2018). Learning regression problems by using classifiers. *Journal of Intelligent & Fuzzy Systems* 35(1), 945–955.
- Atkinson, A. (2020). The box-cox transformation : review and extensions. *LSE Research Online Documents on Economics*.
- Botchkarev, A. (2019). A new typology design of performance metrics to measure errors in machine learning regression algorithms. *Interdisciplinary Journal of Information, Knowledge, and Management* 14, 45–79.
- Boullé, M. (2006). MODL : a Bayes optimal discretization method for continuous attributes. *Machine Learning* 65(1), 131–165.
- Boullé, M. (2007). Compression-based averaging of selective naive Bayes classifiers. *Journal of Machine Learning Research* 8, 1659–1685.
- Breiman, L. (2001). Random forests. *Machine Learning* 45, 5–32.
- Chen, T. et C. Guestrin (2016). Xgboost : A scalable tree boosting system. In *the 22nd ACM SIGKDD International Conference*, pp. 785–794.
- Dua, D. et C. Graff (2017). UCI machine learning repository.

- Fernández-Delgado, M., M. Sirsat, E. Cernadas, S. Alawadi, S. Barro, et M. Febrero-Bande (2019). An extensive experimental survey of regression methods. *Neural Networks* 111.
- Ho, T. K. (1995). Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, Volume 1, pp. 278–282 vol.1.
- Holmes, M. P., A. G. Gray, et C. L. Isbell (2007). Fast nonparametric conditional density estimation. In *Uncertainty in Artificial Intelligence*.
- Janssen, F. et J. Fürnkranz (2010). Separate-and-conquer regression. Technical Report TUD-KE-2010-01, TU Darmstadt, Knowledge Engineering Group.
- Janssen, F. et J. Fürnkranz (2011). Heuristic rule-based regression via dynamic reduction to classification. In T. Walsh (Ed.), *Proceedings of IJCAI*, pp. 1330–1335.
- Langford, J., R. Oliveira, et B. Zadrozny (2012). Predicting conditional quantiles via reduction to classification. *arXiv preprint arXiv:1206.6860*.
- Memon, S. A., W. Zhao, B. Raj, et R. Singh (2019). Neural regression trees. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8.
- Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, et E. Duchesnay (2011). Scikit-learn : Machine learning in Python. *Journal of Machine Learning Research* 12, 2825–2830.
- Rothfuss, J., F. Ferreira, S. Walther, et M. Ulrich (2019). Conditional density estimation with neural networks : Best practices and benchmarks.
- Sondhi, P. (2009). Feature construction methods : a survey. *sifaka. cs. uiuc. edu* 69, 70–71.
- Torgo, L. et J. Gama (1997). Regression using classification algorithms. *Intelligent Data Analysis* 1(1), 275–292.
- Tutz, G. (2021). Ordinal trees and random forests : Score-free recursive partitioning and improved ensembles. *arXiv:2102.00415 [stat.ME]*.

Summary

This paper proposes a method for the automatic creation of variables (in the case of regression) that complement the information contained in the initial input vector. The method works as a pre-processing step in which the continuous values of the variable to be regressed are discretized into a set of intervals which are then used to define value thresholds. Then classifiers are trained to predict whether the value to be regressed is less than or equal to each of these thresholds. The different outputs of the classifiers are then concatenated in the form of an additional vector of variables that enriches the initial vector of the regression problem. The implemented system can thus be considered as a generic pre-processing tool. We tested the proposed enrichment method with 5 types of regressors and evaluated it in 33 regression datasets. Our experimental results confirm the interest of the approach.