Arbres de modèles et flux de données incomplets

Olivier Parisot, Yoanne Didry, Thomas Tamisier, Benoît Otjacques

Luxembourg Institute of Science and Technology (LIST), Belvaux, Luxembourg
olivier.parisot@list.lu, http://www.list.lu

Les arbres de modèles sont des outils puissants pour extraire des modèles prédictifs depuis des données (Quinlan (1992)). En pratique, un algorithme a été proposé pour créer et entraîner itérativement de tels arbres depuis des flux de données (Ikonomovska et Gama (2008)). Ainsi, il est possible de créer et de mettre à jour dynamiquement des modèles de prédiction basés sur des sources constamment alimentées en nouvelles données.

Malheureusement, comme bien souvent en apprentissage automatisé, l'entraînement de ces arbres peut être impacté par des données incomplètes. Par conséquent, il est souvent impératif de pré-traiter les données concernées avant l'apprentissage (Farhangfar et al. (2008)). Mais quels sont les effets sur la taille de l'arbre (*interprétabilité*) et son taux d'erreur (*exactitude*) ?

Afin de contrôler ces aspects, nous proposons un algorithme qui calcule des estimations pour les valeurs manquantes via une méthode d'imputation initiale, puis qui les ajuste afin d'avoir un impact positif sur l'arbre de modèles obtenu. Plus précisément, notre méthode propose de remplacer chaque valeur manquante en utilisant une nouvelle estimation comprise dans un intervalle déterminé par l'estimation de départ et le taux d'erreur courant de la méthode d'imputation. La valeur choisie est celle qui permet à l'arbre de modèles d'avoir le taux d'erreur le plus réduit possible.

Progressivement, durant l'entraînement de l'arbre de modèles, notre méthode permet d'ajuster itérativement les estimations de manière à améliorer le taux d'erreur de l'arbre.

Afin de tester notre approche, un prototype a été développé en JAVA puis intégré dans Cadral, une plateforme interactive d'analyse et de visualisation (Didry et al. (2015)). Il utilise également MOA, une librairie dédiée à l'analyse de flux de données (Bifet et al. (2011)) qui fournit une implémentation de FIMTDD, un algorithme en ligne d'induction d'arbres de modèles (Ikonomovska et Gama (2008)). Le prototype a été testé sur des flux obtenus depuis différentes sources comme le dépot UCI (Bache et Lichman (2013)).

Ensuite, trois approches de prise en compte des valeurs manquantes ont été appliquées sur ces flux de données volontairement incomplets : *a*) Les observations incomplètes sont ignorées. *b*) Les valeurs manquantes sont estimées par la méthode d'imputation. *c*) Les valeurs manquantes sont estimées par la méthode d'imputation puis adaptées par notre algorithme. Dans chacun de ces cas, les métriques suivantes ont été mesurées : *a*) La taille des arbres de modèles obtenus après la phase d'entraînement. *b*) La justesse des arbres de modèles obtenus, calculée sur l'ensemble de validation. *c*) Le taux de confiance de la méthode d'imputation de valeurs manquantes : ce taux est obtenu en comparant les estimations avec les valeurs *réelles* (c'est à dire présentes dans le flux d'origine avant leur suppression pour le test).

Après avoir entraîné des arbres de modèles sur ces flux *artificiellement* incomplets, puis appliqué les trois approches de prise en compte de valeurs manquantes, nous avons observé plusieurs points : *a*) Ignorer les observations incomplètes donne des arbres de modèles plus justes que ceux obtenus en estimant les valeurs manquantes. *b*) Notre méthode d'adaptation des valeurs manquantes permet généralement d'améliorer les résultats. *c*) Notre méthode permet d'obtenir des arbres de modèles globalement plus petits que ceux obtenus avec la méthode initiale d'estimation de valeurs manquantes. Mais en général, ignorer les données incomplètes permet encore une fois d'obtenir de meilleurs résultats, c'est-à-dire des arbres plus compacts. *d*) Si nous comparons les arbres obtenus avec notre méthode d'adaptation et les arbres obtenus avec la méthode initiale d'estimation, nous pouvons constater que l'imputation de valeurs manquantes est positivement impactée.

En conclusion, nous pouvons dire qu'il est globalement plus efficace de ne pas considérer les observations incomplètes durant l'entraînement d'un arbre de modèles depuis un flux de données. Cependant ce n'est pas toujours souhaitable, notamment lorsque des valeurs sont systématiquement/périodiquement manquantes, car cela peut introduire un biais. Dans ce cas, notre méthode d'adaptation des valeurs manquantes permet d'obtenir des résultats plus satisfaisants qu'avec une estimation *standard*.

Dans nos prochains travaux, nous allons appliquer notre méthode sur des flux de données réels issus de capteurs environnementaux. De plus, nous avons dans l'idée d'améliorer notre méthode en mettant au point des heuristiques basées sur les algorithmes génétiques.

Références

Bache, K. et M. Lichman (2013). UCI Machine Learning repository.

Bifet, A., G. Holmes, B. Pfahringer, J. Read, P. Kranen, H. Kremer, T. Jansen, et T. Seidl (2011). MOA : a real-time analytics open source framework. In *Machine Learning and Knowledge Discovery in Databases*, pp. 617–620. Springer.

Didry, Y., O. Parisot, et T. Tamisier (2015). Engineering Data Intensive Applications with Cadral. In *CDVE 2015*. Springer.

Farhangfar, A., L. Kurgan, et J. Dy (2008). Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition 41*(12), 3692–3705.

Ikonomovska, E. et J. Gama (2008). Learning model trees from data streams. In *Discovery Science*, pp. 52–63. Springer.

Quinlan, J. R. (1992). Learning with continuous classes. In *5th Australian joint Conference on Artificial Intelligence*, Volume 92, pp. 343–348. Singapore.

Summary

Model tree is a useful and convenient method for predictive analytics in data streams. Often, this issue is solved by pre-processing techniques applied prior to the training phase of the model. In this article, we propose a new method that estimates and adjusts missing values before the model tree training. A prototype was developed and tested on several data streams.