Vers un échantillonnage de flux de données transformé

Olivier Parisot, Thomas Tamisier

Luxembourg Institute of Science and Technology, Belvaux, Luxembourg
olivier.parisot@list.lu

De nombreuses techniques ont été mises au point récemment afin d'extraire des modèles prédictifs depuis des flux de données (Nguyen et al., 2015). Dans ce domaine, le calcul de l'exactitude des résultats est crucial pour évaluer la performance des modèles obtenus. Avec des flux potentiellement infinis, il faut donc maintenir un échantillonnage, ce dernier étant utilisé – à intervalle régulier ou à la demande – pour calculer le taux d'erreur courant sur un ensemble représentatif du flux (par ex., un mélange bien balancé entre éléments récents/anciens).

Dans ce papier, nous appliquons l'échantillonnage durant l'entraînement d'un modèle prédictif et nous le transformons afin de générer à la demande un arbre de décision simplifié montrant quand ce modèle se trompe. Pour se faire, notre méthode se décline de manière classsique en une phase *online* et en une phase *offline* (Fig. 1).

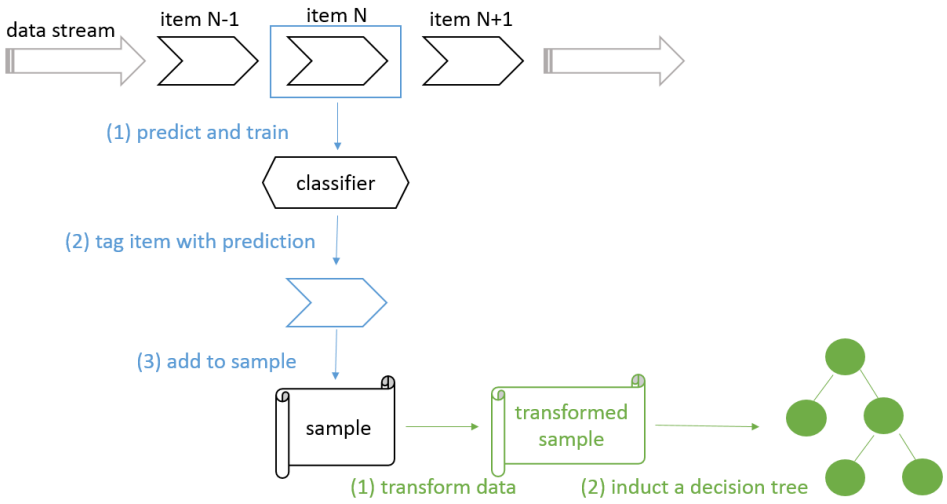

FIG. 1 – *Approche pour créer à la demande un arbre de décision montrant les erreurs d'un modèle de prédiction entraîné sur un flux (phase online en bleue, phase offline en vert).*

Durant la phase *online*, chaque élément du flux est analysé : il est évalué par le modèle prédictif en cours, tagué en fonction de la justesse de la prédiction ('*bien classé* / *mal classé*'), puis inséré dans un échantillonnage (de type réservoir, par exemple – Vitter (1985)).

Durant la phase *offline*, l'échantillonnage en cours est utilisé pour créer un arbre de décision (la classe de cet arbre étant l'attribut '*bien classé / mal classé*'). Ainsi nous obtenons un arbre qui explique quand les éléments ont été bien prédits par le modèle. L'arbre généré pouvant être grand et donc dfficilement interprétable (Luštrek et al., 2016), nous avons appliqué une phase intermédiaire de transformation de l'échantillonnage afin de le simplifier (i.e. réduire la taille). Pour se faire, nous avons adapté et intégré un algorithme génétique permettant de déterminer une séquence d'opérations pour modifier les données (normalisation, discrétisation, fusion de champs, etc.) (Parisot et al., 2014). Cet algorithme, initialement utilisé pour réduire les arbres de régression, est exécuté durant la phase *offline* et donc ne pénalise pas l'analyse *online*.

Nous avons développé un prototype Java basé sur Weka et MOA (Bifet, 2015), et nous avons utilisé : *a*) VFDT comme algorithme d'apprentissage de modèle prédictif (Hulten et al., 2001), *b*) l'échantillonnage par réservoir pour stocker les prédictions récentes (Vitter, 1985). *c*) J48 pour générer l'arbre à la demande depuis l'échantillonnage courant.

Par exemple, l'approche a été appliquée pour créer un modèle prédictif à partir du volumineux jeu de données artificiel *CovPokElec* (73 attributs, 1455525 éléments, classe avec 10 valeurs possibles). Pendant l'analyse (540000ème élément, plus précisément), pour un échantillonnage de type réservoir de taille 500, nous avons constaté que le taux d'erreur du modèle est de 32 %. A cet instant, pour observer quand les prédictions sont bonnes ou mauvaises, cet échantillonnage est transformé avec notre adaptation de l'algorithme génétique proposé par (Parisot et al., 2014) en vue d'obtenir un arbre simplifié (au format textuel Weka) :

```
Soil_Type40==0
| Soil_Type39==0
| | Soil_Type24==0: GOOD PREDICTION
| | Soil_Type24==1: BAD PREDICTION
| Soil_Type39==1: BAD PREDICTION
Soil_Type40==1: BAD PREDICTION
```

Lors de nos futurs travaux, nous améliorerons notre solution pour gérer les cas où les arbres compactés ont encore un nombre de noeuds importants (ex : 50,100), en ayant recours à des techniques de visualisation plus adaptées (Luštrek et al., 2016).

Références

Bifet, A. (2015). Real-time big data stream analytics. In *2nd Annual International Symposium on Information Management and Big Data*, pp. 13.

Hulten, G., L. Spencer, et P. Domingos (2001). Mining time-changing data streams. In *ACM SIGKDD 2001*, pp. 97–106. ACM.

Luštrek, M., M. Gams, S. Martinčić-Ipšić, et al. (2016). What makes classification trees comprehensible ? *Expert Systems with Applications 62*, 333–346.

Nguyen, H.-L., Y.-K. Woon, et W.-K. Ng (2015). A survey on data stream clustering and classification. *Knowledge and information systems 45*(3), 535–569.

Parisot, O., Y. Didry, et T. Tamisier (2014). Data wrangling : A decisive step for compact regression trees. In *CDVE 2014*, pp. 60–63. Springer.

Vitter, J. S. (1985). Random sampling with a reservoir. *ACM Trans. Math. Softw. 11*(1), 37–57.