Revue littéraire - CART & confidentialité différentielle

Thomas

September 2017

1 Introduction

Article ici.

Approche 1 : Former un arbre de décision respectant la confidentialité des données en se basant sur l'algorithme "classique" ID3 grâce à des requêtes respectant la confidentialité différentielle.

Approche 2 : Utilisation d'un ensemble de classifieurs basés sur la confidentialité différentielle. Ici, on prend comme classifieurs des arbres de décisions aléatoires.

1.1 CART basé sur ID3

Les résultats sont médiocres, puisqu'il est impossible d'obtenir à la fois un classifieur ayant une bonne précision et une bonne valeur de confidentialité, en particulier sur les petits jeu de données. La qualité des prédictions est d'autant plus altérées que l'on est obligé de prendre un paramètre de confidentialité (ε) faible afin de pouvoir maintenir l'anonymisation à un niveau raisonnable.

Un paramètre de confidentialité ε faible induit une confidentialité élevée - cf. Partie 2.1

1.2 Ensemble d'arbres de décisions aléatoires

Cette méthode bénéficie d'une bonne précision tout en préservant le caractère confidentiel des données, même sur les petits jeu de données.
Cependant et contrairement aux arbres de décisions élaborés à partir de l'algorithme ID3, les arbres décisionnels aléatoires se comportent comme des boîtes noires. Il n'est donc pas recommandé d'utiliser ces algorithmes afin d'essayer de comprendre quelles combinaisons attributs-valeurs ont le plus grand pouvoir prédictif.

2 Rappels et définitions

2.1 Confidentialité différentielle

Soit $\mathcal{D}_1, ...\mathcal{D}_k$ des espaces pouvant être numériques et catégoriels. Une base de données D est composé de n lignes : $\{x_1, ..., x_n\}$, tel que $x_i \in \mathcal{D}_1 \times ... \times \mathcal{D}_k$.

Définition. Un algorithme $\mathcal{M}$ vérifie le concept de ε-confidentialité différentielle, si $\forall D_1, D_2$ différant au plus d'un élément et $S \in \mathrm{Im}(\mathcal{M})$, on a :

$$\mathbb{P}_{\mathcal{M}}[\mathcal{M}(D_1) = S] \leq e^{\epsilon}.\mathbb{P}_{\mathcal{M}}[\mathcal{M}(D_2) = S]$$

On remarque donc qu'une valeur petite de ε induit des distributions proches et donc une confidentialité plus importante.

Définition. La sensibilité globale de la fonction f est le plus petit nombre $S(f)$ tel que $\forall D_1, D_2$ différant d'une seul élément, on a :

$$\| f(D_1) - f(D_2) \|_1 \leq S(f)$$

Théorème. Soit f une fonction à valeur dans $\mathbb{R}^n$. L'algorithme retournant $f(D) + Y_1 + ... + Y_m$, où Y_i est une variable aléatoire iid selon la loi $\text{Lap}(S(f)/\epsilon)$ satisfait le principe de ε-confidentialité différentielle.

Puisque la variance de Y_i vaut $2(\frac{S(f)}{\epsilon})^2$, on remarque que plus ε est petit, plus le bruit ajouté à la valeur de retour de l'algorithme est important et donc plus le caractère confidentiel est préservé.

Théorème.(*Composition successive*) Une séquence successive d'algorithmes (ou de requêtes) $\mathcal{M}_i$, chacun ϵ_i-différentiellement confidentiel , est $\sum_i \epsilon_i$-différentiellement confidentiel.

Théorème. (*Composition parallèle*) Soient $\mathcal{M}_i$ des algorithmes chacun ϵ_i-différentiellement confidentiels, et D_i des ensembles disjoints de D. Alors le processus exécutant chaque $\mathcal{M}_i$ sur chaque D_i en parallèle est ϵ_i-différentiellement confidentiel.

Exemple : On considère une fonction f tel que $S(f) = 1$. On prend une valeur de confidentialité $\epsilon = 0.01$, l'écart-type pour le bruit issu de la distribution de Laplace est d'environ 141 pour une unique requête $\mathcal{M}$, tel que $\mathcal{M}$ est 0.01-différentiellement confidentiel.
En décidant de réaliser successivement q requêtes $\mathcal{M}_i$ via l'algorithme $\sum \mathcal{M}$, on peut écrire que $\sum \mathcal{M}$ est $0.01q$-différentiellement confidentiel. Or $q > 1$, donc la qualité de la confidentialité de cet algorithme est détérioré par rapport à la confidentialité d'une unique requête $\mathcal{M}_i$.
Si l'on souhaite avoir l'algorithme $\sum \mathcal{M}$ 0.01-différentiellement confidentiel, on doit choisir ϵ_i, pour chaque requête $\mathcal{M}_i$, tel que :

$$\epsilon_i = \frac{0.01}{q} \Rightarrow Stdv \approx \frac{1.41}{0.01q} \approx 141q$$

Il faut donc augmenter substantiellement la quantité de bruit à insérer, à tel point que sur une "petite" base de données le bruit pourrait totalement recouvrir le "signal" initial.

2.2 Arbre décisionnels ID3

Il s'agit d'un algorithme récursif descendant afin de calculer un classifieur sous forme d'arbre de décision à partir de données "étiquetées". Il s'agit à ce titre d'un algorithme supervisé.
L'idée de cet algorithme consiste à scinder le jeu de données d'apprentissage selon des critères (conditionnement sur un ou des couples attributs-valeurs) maximisant le gain d'entropie résultant. Une fois la totalité de l'échantillon d'apprentissage séparé selon les différentes règles de décisions, une phase de d'élagage visant à supprimer les feuilles et les branches "inutiles" de l'arbre est réalisé afin de minimiser au maximum le risque de sur-apprentissage.

2.3 Arbres de décisions aléatoires

L'idée ici est d'utiliser une méthode ensembliste basée sur des arbres de décisions aléatoires.
Les attributs pour chaque noeud des arbres sont sélectionnés aléatoirement, à la différence de l'algorithme ID3 où leurs choix s'effectuaient selon des critères particuliers (entropie, coefficient de gini,...).
La structure de l'arbre (le choix des attributs au sein des différents noeuds) est d'ailleurs définie avant que les données ne soient analysées. Elles seront examinées seulement pour modifier l'arbre et étiqueter ses feuilles.

Algorithme pour créer un arbre aléatoire :

(1) Déterminer récursivement la structure de l'arbre.

(2) Mettre à jour les statistiques de l'arbre, à partit des données d'apprentissage, en comptant pour chaque feuille le nombre d'instance de chaque classe que l'on y trouve.

(3) Supprimer les noeuds et les feuilles dans lesquels on ne trouve aucune instance d'aucune classe[1].

[1]Pour les valeurs continues, il suffit de choisir un seuil aléatoire pour chaque attribut.

Il y deux paramètres importants dans la configuration de cette méthode, à savoir la hauteur des arbres h (dont Fan et al. détermine une valeur optimale tel que $h = m/2$, avec $m =$ le nombre d'attributs), ainsi que N le nombre total d'arbres aléatoires à calculer pour obtenir notre classifieur ensembliste final (une valeur aussi petite que 10 suffit).

Avantages :

(+) L'efficacité de la phase d'apprentissage : il suffit d'une unique passe sur les données par arbre crée.

(+) L'algorithme pour construire chaque arbre réalise un faible nombre de requêtes sur la base de données, ce qui permet de nous maintenir au sein du cadre de la confidentialité différentielle.

(+) Dans une moindre mesure, le faible espace mémoire nécessaire pour son éxecution.

3 Arbres de décisions ID3 différentiellement confidentiel

La plupart des travaux réalisés dans le cadre de la confidentialité différentielle repose sur l'insertion de bruit au sein des réponses de requêtes de bas-niveaux (ex: *compter le nombre d'instances qui vérifient une certaine condition*).
Ces requêtes de bas-niveaux peuvent être employée afin de construire des algorithmes de machine learning tel que des arbres de décisions. Néanmoins comme nous l'avons vu lors du théorème de composition successive de requêtes bas-niveau afin d'établir une structure de haut-niveau différentiellement confidentielle : la confidentialité globale obtenue est bien inférieure à celle des requêtes bas-niveaux utilisées (cf. *exemple*).
Il faut alors nécessairement diminuer le niveau de confidentialité des requêtes bas-niveaux afin d'avoir une confidentialité globale acceptable, au détriment d'un impact négatif sur la précision de la structure de haut-niveau ainsi obtenue, allant même jusqu'à finalement remettre en cause son intérêt.
Par ailleurs plusieurs changements seront bien entendu nécessaire lors de l'implémentation de l'algorithme afin le rendre compatible avec les contraintes liées à la confidentialité différentielle :

- Remplacer les conditions sur la taille de l'échantillon du jeu de données d'apprentissage par une requête qui compte le nombre d'éléments au sein de la base.
- Remplacer les conditions sur les valeurs des classes des instances du jeu de données, par une requête qui compte le nombre d'éléments de chaque classe.
- Lors du calcul d'entropie, remplacer l'évaluation de la taille des sous-ensemble du jeu d'apprentissage contenant certaines valeurs de la classe cible par des requêtes de sensibilité globale égale à 1.

A noter que l'on peut utiliser le théorème de composition parallèle pour les requêtes sur un même niveau de l'arbre puisque les sous-ensemble de chaque noeud d'un même niveau sont bien disjoints, cependant nous sommes obligés de recourir au théorème de composition successives pour les requêtes sur des niveaux différents.
Malheureusement, le nombre de requêtes inter-niveaux s'avère être assez important conduisant à choisir un paramètre de confidentialité (très) faible pour chacunes de ces requêtes bas-niveaux ce qui explique les mauvais résultats de l'algorithme (*cf. table 1*).

A noter que la valeur du paramètre de confidentialité global choisie ϵ' est de 0.5, ce qui implique que la valeur de confidentialité des requêtes bas-niveaux ϵ soit égale à ϵ'/q, où q est le nombre maximum de requêtes réalisées entre les niveaux de l'arbre. Cependant q ne peut être connu à l'avance, on considère donc que le nombre maximum de requêtes entre les niveaux ne peut excéder le nombre d'attributs de la base de données.

Table 1: *Implémentation d'un arbre de décision ID3 préservant la confidentialité des données s'avérant être un piètre classifieur* :

Données	**# lignes**	**# requêtes**	**Précision ID3 original**	**Précision ID3 confidentiel**
Nursery	12960	36	98.19%	46%
Cong. votes	435	136	94.48%	40%
Mushroom	8124	253	100%	56%

4 Arbres de décisions aléatoires différentiellement confidentiel

4.1 Méthodologie

On considère ici une version modifiée de l'algorithme présenté précédemment :

(1) Suppression de l'étape d'élagage

(2) Remplacement du comptage de chaque classe dans chaque feuille par des requêtes différentiellement confidentiel.

Grâce à la suppression de l'élagage, la structure de l'arbre ne dépend désormais plus du tout des données. Par ailleurs toutes les feuilles des arbres se trouvent également au même niveau.
On peut donc maintenant définir un vecteur feuille V de dimension $(M \times T)$, avec M le nombre de feuille que contient l'abre et T le nombre de labels différents pour la classe cible. Ce vecteur fait donc également office de comptabilisateur des différente instances.
Il suffit donc pour construire un arbre de décision aléatoire de retourner :

(1) La structure de l'arbre

(2) le vecteur de feuille V

On montrera par la suite que le vecteur V possède une sensibilité de 1, et qu'en ajoutant un bruit suivant une loi $\text{Lap}(1/\epsilon)$ à chacune de ses composantes avant de le retourner on obtient un vecteur ε-différentiellement confidentiel.
On répète alors autant de fois que nécessaire ces étapes afin d'obtenir le nombre escomptés d'arbres de décisions aléatoires différentiellement confidentiel. On obtient alors la création d'un enemble de N RDT en ajoutant un bruit $\sim \text{Lap}(N/\epsilon)$ à chaque vecteur feuille, avec une complexité $O(n\log(n))$ pour construire un unique arbre, où n est la taille du dataset.

Choix de la hauteur optimale :

$$h = \min(\lfloor k/2 \rfloor, \lfloor \log_b(n) \rfloor - 1)$$

avec, b le nombre moyen de valeurs prises par les attributs ($\sim$ mean branching factor), k le nombre d'attributs et n la taille du dataset.

4.2 Mise à jour de l'arbre

L'algorithme précédent nécessite d'avoir toutes les données disponibles au départ lors de la construction de l'arbre.
Dans la réalité, il est fréquent d'obtenir les données par vagues, il faut donc pouvoir "mettre à jour" le classifieur existant avec ces nouvelles données, en ayant recours à l' *incremental learning*.

Problème : Incrémenter plusieurs fois le RDT dans un cadre de confidentialité différenciée peut conduire à une diminution de la protection de l'anonymat (*cf. théorème de composition successive*).

L'article envisage une possibilité de résoudre ce problème en copiant la structure de l'arbre construit sur les données initiales puis en y insérant les nouvelles données et en recalculant le nouvel vecteur feuille obtenu. Ajouter à ce vecteur un bruit Laplacien puis le sommer au vecteur feuille initial afin de pouvoir retourner le vecteur feuille ainsi formé ainsi que la structure de l'arbre mise à jour. En contrepartie, on note une diminution de la précision du classifieur obtenu comparé à un classifieur crée directement à partir des données combinées.

4.3 Arbre de décision aléatoires privés à partir de bases de donnés distribuées

L'enjeu consiste à pouvoir construire un RDT à partir de bases de données partitionnées horizontalement et verticalement :

4.3.1 Partitionnement horizontal

L'idée pour obtenir un tel classifieur à partir d'un dataset partitionné selon les individus (et où chaque partition détient les mêmes variables mais sur des individus tous strictement distincts) consiste à reprendre l'algorithme de mise à jour en considérant les différentes partitions comme des vagues de nouvelles données.

4.3.2 Partitionnement vertical

Ici chaque partition possède la totalité des individus mais sur des variables toutes strictement distinctes, sauf une commune pour effectuer la jointure. Il suffit donc de construire les arbres en appliquant l'algorithme précédent sur chacune des k partitions, puis de considérer les k classifieurs obtenus au sein d'un unique nouveau classifieur où seront réalisées les prédictions.

Dans chacun des cas (partitionnement horizontal et vertical) les classifieurs sont formés à partir de jeux de données disjoints, donc l'union de ces classifieurs est également différentiellement confidentiel.

5 Résultats d'expériences

Les jeux de tests ont été réalisés sur trois datasets différents dont leurs tailles varient, allant de 12 000 à 400 lignes seulement.
Les tests ont également été réalisés avec plusieurs valeurs de ε, tel que $\epsilon = \{\infty; 5; 4; 3; 2; 1; 0.75; 0.5; 0.25; 0.1; 0.01\}$. ($\epsilon = \infty \Rightarrow$ pas de confidentialité).
Les résultats obtenus correspondent à une moyenne de dix exécutions des algorithmes, à la fois ID3 et RDT différentiellement confidentiel.

5.1 Précision

Globalement la performance du classifieur par arbres de décisions aléatoires différentiellement confidentiel est assez constantes quelque soit les jeux de données utilisés pour les tests (*précision* $\approx 90\%$ jusqu'à $\epsilon = 0.5$).
Par ailleurs l'écart de performance entre les algorithmes ID3 et RDT diminue lorsque la taille de la base de données augmente.

5.2 Performance du partitionnement horizontal

Les tests ont été réalisés avec différents nombre de "mises à jour", les résultats obtenus soulignent que la précision diminue avec le nombre de mises à jour réalisées.

5.3 Performance du partitionnement vertical

En se basant sur un nombre fixé de partitions (2), on remarque que la précision diminue lorsque ϵ augmente, alors que la variance elle aussi augmente.

References

[1] G. Jagannathan, K. Pillaipakkamnatt, R.N Wright, *A Practical Differentially Private Random Decision Tree Classifier.* , 2009.