Sécurité Appliquée-TP 2
Protection de la vie privée-PVP

Jean-François COUCHOT
couchot[arobase]femto-st[point]fr

6 novembre 2020

Ce TP s'inspire largement de l'article de J. Brickell [1] critiquant les approches syntaxiques à base de de k-anonymat et leurs dérivées.

L'idée principale est de comparer différents niveaux de protection de la vie privée (du plus stricte à des versions plus allégée) en terme d'apprentissage. Le gain en précision d'apprentissage vaut-il le coût qu'il impose quant à la fuite d'informations personnelles ? C'est la question auquelle ce TP apportera une réponse.

1 Initialisation du TP

On considère le jeu de données Adults de l'UCI présentant un extrait des données de recensement en 1994/1995 aux USA et dont l'objectif initial était de prédire si telle ou telle personne allait avoir un salaire supérieur à 50K$ par an. Les attributs que l'on va conserver sont :

— âge (*age*)
— catégorie professionnelle (workclass) :
— éducation (*education*) : niveau d'éducation
— statut marital (marital-status)
— origine ethnique (race)
— genre (sex)
— pays d'origine (native country) : où la personne est née
— secteur de travail (*occupation*) : dans quel secteur travaille la personne

Exercice 1.1. *Faire le nettoyage dans un tableur pour ne conserver que les attributs ci-dessous et enregistrer le tout dans le format csv.*

1.1 Le meilleur apprentissage possible

On imagine ici que la/le scientifique responsable de l'analyse a accès à toutes les données, c.-à-d. qu'aucune donnée n'est supprimée, ni transformée. C'est le cas le plus favorable pour elle/lui.

Exercice 1.2. *A l'aide de l'outil weka, essayer de faire apprendre le statut marital, en utilisant les trois approches de classification avec les paramètres par défaut pour J48, Random Forests, and Naive Bayes. Quelle est le score maximal en termes d'apprentissage (U_{Max}) ?*

1.2 L'apprentissage le plus respectueux de la vie privée

On imagine que la confiance n'est pas de mise. Tous les quasi-identifiants ont été généralisés à l'extrême et l'on va rejouer l'apprentissage, pour évaluer la précision de l'approche, à minima. Se pose immédiatement la question de définir les quasi-identifiants. Dans ce TP on considérera que les attributs *age*, *education*, *occupation* sont les quasi-identifiants.

Exercice 1.3.
1. *Générer un jeu de données où tous les quasi-identifiants ont été généralisés à l'extrême. Quelle est alors la valeur de Loss ? Exporter ce jeu de données.*
2. *Sur ce jeu de données, réaliser le même apprentissage que dans la section précédente. Quel est le score minimal (U_{Min}) ?*

1. Brickell, J., & Shmatikov, V. (2008, August). The cost of privacy : destruction of data-mining utility in anonymized data publishing. In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 70-78).

2 Des nettoyages syntaxiques

2.1 Des nettoyages utiles pour l'apprentisage, mais respectueux de la vie privée ?

Les quasi-identifiants ci-dessus vont être généralisés selon les hiérarchies suivantes :
— age : regroupement par intervalles de taille 10, 20, 40, 80, à partir de 17.
— éducation : regroupement dans les ensembles suivants : {Preschool, 1st-4th, 5th-6th}, {7th-8th, 9th}, {10th, 11th, 12th, HS-grad}, {Some-college, Assoc-voc, Assoc-acdm, Bachelors}, {Masters, Prof-school, Doctorate} puis généralisation globale.
— occupation : généralisation globale.

Exercice 2.1. *Implanter ces hierarchies.*

Exercice 2.2 (Un apprentissage à base de 10-anonymat).
1. *Générer un jeu de données respectueux du 10-anomnymat en acceptant jusqu'à 5% de suppression. Exporter ce jeu de données.*
2. *Sur ce jeu de données, réaliser le même apprentissage que dans la section précédente. Quel score ($U_{k_{10}}$) optenez vous ? En pourcentage, exprimez l'amélioration par rapport à U_{Min}*
3. *Calculer le gain d'inforamtion moyen $\mathcal{A}_{know}$ pour un adversaire suite à la publication de ce jeu de données.*
4. *Conclure.*

Exercice 2.3 (Un apprentissage à base de 2-diversité). *Reprendre l'exercice précédent en changeant le modèle précédent par de la 2-diversité. Ne pas oublier de conclure*

2.2 Un nettoyage respectueux de la vie privée, utile pour l'apprentissage ?

Exercice 2.4 (Un apprentissage à base de 0.3-proximité).
1. *Reprendre l'exercice 2.2 en prenant comme modèle la 0.3-proximité. Ne pas oublier de conclure.*
2. *Donner une conclusion générale quant à la protection de la vie privée assurée par des méthodes syntaxiques (k-anonymat, l-diversité, t-proximité).*

3 Ajouter du bruit borné

On considère une base D de n individus qui ont (1) ou pas (0) une maladie et un algorithme de comptage bruité Q'.

Exercice 3.1 (Mise en place du TP d'ajout de bruit borné).
1. *Générer une base D (de taille $n = 15$ par exemple) de bits choisis aléatoirement.*
2. *Développer Q' qui ajoute un bruit selon une loi normale centrée d'écart type 3α et borné par α.*

Exercice 3.2 (Recontruire une base par brute force).
1. *Pour chacune des requêtes possible S ($s \in \{0,1\}^n$), stocker dans une liste les valeurs de $Q'_S(D)$ puisque celles-ci sont connues de l'attaquant.*
2. *Développer l'algorithme qui permet de construire la base D'*
   — *Ici l'espace de recherche est fini puisqu'il y a n individus avec comme valeurs 0 ou 1 ; Ainsi $D' \in \{0,1\}^n$.*
   — *Pour chaque D', il suffit d'évaluer $|Q_S(D') - Q'_S(D)| < \alpha$ pour chaque requête S.*
3. *Afficher l'erreur commise $e = \|D - D'\|_1$.*
4. *Appliquer la méthode pour $\alpha = n/10$, $n/2$, n pour $n = 15,\ldots, 20$.*
5. *Montrer que l'algorithme d'attaque est en $\mathcal{O}(2^{2n})$.*
6. *Que conclure ainsi si l'attaquant a accès à toutes les requêtes, même bruitées ?*

Exercice 3.3 (Recontruire une base par par programmation linéaire). *L'attaque présentée dans l'exercice précédent n'est pas viable sur des bases de grande taille. Cette approche nécessite d'avoir acccès à toutes les requêtes possibles, soit un nombre de 2^n.*

Dans cet exercice, on s'intéresse à attaquér la base avec beaucoup moins de requêtes $n \times (\log(n))^2$.

1. *Générer une base D (de taille $n = 40$ par exemple).*

2. *Développer l'algorithme qui permet de trouver une base D' proche de D mais cette fois basé sur la programmation linéaire.*
   - *Toutes les reqûetes ne vont pas être construites : choisir aléatoirement $t = n \times (\log(n))^2$ sous ensembles distincts de $\mathcal{X}$. qui seront les t requêtes sur lesquelles l'évaluation sera faite.*
   - *Pour chacune de ces requêtes, stocker dans une liste les valeurs de $Q'_S(D)$.*
   - *Soummetre à un solveur de programmation linéaire sur le vecteur d'inconnues $(D'_1, \ldots, D'_n)$ le problème suivant :*
   $$|S.D' - Q'_S(D)| \leq \alpha \text{ pour chaque } S$$
   $$0 \leq D'_i \leq 1$$
   - *Arrondir les D'_i à l'entier le plus proche (0 ou 1).*
3. *Afficher l'erreur commise $e = \|D - D'\|_1$.*
4. *Appliquer la méthode pour $\alpha = 1, \ldots, \sqrt{n}, \ldots, n/2$, pour $n = 40, \ldots, 50$.*
5. *Que conclure ainsi si l'attaquant a accès à $n \times (\log(n))^2$ requêtes, même sur des bases de plus grande taille, pour un bruit inférieur à $\sqrt{n}$?*