Algorithmes stochastiques

Devoir à la maison, à rendre le dimanche 8 novembre 2020 au plus tard

Ce devoir est à rendre en ligne en remplissant ce formulaire :

https://forms.gle/8nGu22WT7nDj8vuLA

Veuillez indiquer sur ce formulaire, en plus de vos noms et emails, des liens (en ligne, sur le cloud de votre choix) vers deux fichiers composant votre devoir :

- **La partie théorique sous forme d'un fichier pdf (les notes manuscrites scannées sont acceptées) que vous appellerez 'DMalgosto-votrenom.pdf';**
- **La partie informatique sous forme d'un Jupyter Notebook indépendant que vous appellerez 'DMalgosto-votrenom.ipynb'.**

Ne m'envoyez pas votre devoir par email !

Ce devoir est individuel, le soin et l'originalité que vous apporterez dans vos réponses compteront dans la note. Toute ressemblance trop forte entre deux devoirs sera sanctionnée.

1 Sujet - partie théorique

Dans ce devoir, on suppose qu'une professeure de math a posé un examen à ses étudiants dans lequel chaque question a une réponse de type "Vrai ou Faux" (1 ou 0). Au moment de corriger ses copies, l'enseignante s'aperçoit qu'elle a oublié quelles étaient les bonnes réponses...

Elle a N copies à corriger (donc N étudiant.es), et l'examen comporte M questions.

On note $x_{n,m} \in \{0,1\}$ la réponse (**connue**) de l'étudiant.e n à la question m, et $\mathbf{x} = (x_{n,m})_{\substack{n=1,\dots,N \\ m=1,\dots,M}}$ l'ensemble de toutes leurs réponses aux questions.

Le but de la correction est d'estimer une note (ou un score) θ_n pour chaque étudiant.e n, et de retrouver également les vraies réponses $\mathbf{t} = (t_m)_{m=1,\dots,M}$ (**inconnues**) aux questions, et tout cela

seulement à partir de l'ensemble $\mathbf{x}$ des réponses des étudiant.es aux questions. Chaque réponse t_m vaut 0 ou 1.

Remarquons que si on connaissait les vraies réponses $(t_m)_{m=1\dots M}$, alors l'étudiant.e numéro n aurait une réponse juste à la question m si et seulement si $x_{n,m} = t_m$, et on pourrait donc en déduire une note $\theta_n \in [0, 1]$ pour l'étudiant.e numéro n en comptant sa moyenne de réponses justes.

On propose donc de modéliser ce problème d'estimation de manière similaire à celui étudié pour les mélanges de gaussiennes, où l'on a des observations (les réponses $x_{n,m}$) qui dépendent de paramètres (les notes θ_n) et de variables latentes binaires (les vraies réponses aux questions t_m).

On va utiliser la modélisation suivante :

- les réponses binaires t_m sont des réalisations de variables T_m indépendantes de même loi de Bernoulli de paramètre 0.5 ($\mathbb{P}(T_m = 1) = \mathbb{P}(T_m = 0) = 0.5$).
- les réponses $x_{n,m}$ sont des réalisations de variables aléatoires indépendantes $X_{n,m}$, et la variable $X_{n,m}$ suit la loi conditionnelle suivante

$$\mathbb{P}[X_{n,m} = T_m | T_m] = \theta_n \ \text{, et } \mathbb{P}[X_{n,m} = 1 - T_m | T_m] = 1 - \theta_n.$$

On note $\theta = (\theta_1, \dots, \theta_N)$ l'ensemble des notes (**inconnues**) des étudiant.es. On souhaite écrire un algorithme permettant d'estimer le vecteur de notes θ et le vecteur des réponses aux questions $\mathbf{t} = (t_m)_{m=1\dots M}$, à partir des observations $\mathbf{x}$.

1. Pour initialiser notre algorithme, on propose d'initialiser les réponses t_m en supposant que pour chaque question m, une majorité d'étudiant.es répondent correctement à la question. Ecrire les estimateurs $\hat{t}_m$ correspondant à cette hypothèse.
2. Montrer que la log-vraisemblance complète du problème d'estimation peut s'écrire

$$\log p(\forall m,\ T_m = t_m,\ \text{et } \forall n, m\ X_{n,m} = x_{n,m} | \theta) = M \log 0.5 + \sum_{n=1}^{N} \sum_{m=1}^{M} \log \left(\theta_n^{\mathbb{1}_{x_{n,m}=t_m}} (1 - \theta_n)^{\mathbb{1}_{x_{n,m}=1-t_m}} \right).$$

3. Montrer que si a et b valent 0 ou 1, alors $\mathbb{1}_{a=b} = ab + (1 - a)(1 - b)$.
4. Dériver l'expression de la log-vraisemblance par rapport aux notes θ_n. En déduire un estimateur $\hat{\theta}_n$ de la note de l'étudiant.e numéro n lorsque les réponses t_m sont connues. Ecrire cet estimateur $\hat{\theta}_n$ comme une fonction linéaire des t_m grâce à l'expression montrée à la question précédente.
5. En pratique, on ne connaît pas les réponses t_m, donc on ne peut pas calculer $\hat{\theta}_n$ directement par maximum de vraisemblance. On va utiliser à la place des t_m leurs espérances connaissant l'ensemble des observations $\mathbf{x}$. Montrer que si θ est fixé, alors

$$\mathbb{E}[T_m | \mathbf{x}, \theta] = \frac{\prod_{n=1}^{N} \theta_n^{x_{n,m}} (1 - \theta_n)^{1 - x_{n,m}}}{\prod_{n=1}^{N} \theta_n^{x_{n,m}} (1 - \theta_n)^{1 - x_{n,m}} + \prod_{n=1}^{N} \theta_n^{1 - x_{n,m}} (1 - \theta_n)^{x_{n,m}}}. \tag{1}$$

6. En déduire un algorithme complet de type Espérance-Maximisation pour estimer les notes $\theta = (\theta_1, \ldots, \theta_n)$ et les variables latentes $t_1, \ldots, t_m$ (les vraies réponses aux questions). Ecrire le pseudo-code de cet algorithme et expliquer quelle étape correspond à une maximisation et quelle étape à une espérance.

2 Partie informatique

Dans cette partie, il est demandé de coder l'algorithme précédent en Python. Cette partie sera à rendre sous la forme d'un Jupyter Notebook qui puisse être ouvert et lancé indépendamment. Le notebook ne devra faire appel qu'à la librairie Numpy.

Pour tester l'algorithme, deux fichiers de données 'x.npy' et 't.npy' sont à télécharger sur la page du cours. Vous pouvez les ouvrir avec Python (en les mettant dans le même répertoire que votre code) avec les commandes `x = np.load('x.npy')` et `t = np.load('t.npy')`. Le tableau `x` de taille $N \times M$ contient les notes des élèves, et le tableau `t` de taille $1 \times M$ est le tableau des vraies réponses aux questions. Ce tableau `t` est fourni uniquement pour que vous puissiez vérifier que votre algorithme fonctionne bien : votre code doit permettre de retrouver `t` à partir de la seule donnée `x`.

Les remarques suivantes sont à prendre en compte pour cette partie du devoir :

- L'algorithme complet doit comporter :
  1. une initialisation de $\mathbf{t}$ (voir la question 1 de la partie théorique)
  2. I itérations (prendre $I = 10$) alternant les estimations de θ et de l'espérance de $\mathbf{t}$
- L'algorithme prend en entrée la matrice Numpy `x` de taille $N \times M$ contenant les réponses aux questions des étudiant.es, et doit retourner la valeur estimée du vecteur de notes θ, ainsi qu'une estimation de $\mathbf{t}$.
- Comme l'algorithme ne calcule pas $\mathbf{t}$ (qui est un vecteur de 0 et de 1) mais son espérance, on propose de seuiller cette espérance à 0.5 pour obtenir une estimation de $\mathbf{t}$. A la fin de l'algorithme, pour tous les indices m pour lesquels l'espérance est supérieure à 0.5, t_m est mis à 1, et pour les autres t_m est mis à 0.
- Lorsque a et b sont deux valeurs positives mais très petites, calculer $\frac{a}{a+b}$ n'est pas toujours stable numériquement. En posant $A = \log(a)$, $B = \log(b)$, montrer qu'on peut remplacer ce calcul par
$$e^{A-\log(e^A+e^B)}.$$
Expliquer l'intérêt de cette démarche. On l'utilisera pour implémenter le calcul de (1) dans l'algorithme.