Examen

Master parcours SSD - UE Statistique Computationnelle

Automne 2022

Consignes : *Cet examen compte pour la moitié de l'UE. Il prend la forme d'un devoir maison pour lequel vous devrez me rendre :*

1. *un rapport décrivant les analyses effectuées et commentant les résultats obtenus,*
2. *le(s) script(s)* `R` *correspondant,*

Il est à réaliser **en binôme pour le 14 novembre** *et à soumettre sur moodle (un rendu par binôme).*

1 Exercice 1 - simulation de la loi normale centrée réduite

L'objectif de cet exercice est d'implémenter une procédure de simulation de la loi normale centrée réduite en combinant les méthodes d'inversion et de rejet. Formellement, on souhaite donc simuler un échantillon distribué selon la densité $f(x) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-x^2}{2}\right)$, pour $x \in \mathbb{R}$. On mettra pour cela en oeuvre une procédure de rejet, en s'appuyant sur la densité auxiliaire $g(x) = \frac{1}{2}\exp\big(-|x|\big)$. Comme on ne dispose pas de fonction `R` permettant de simuler selon $g(x)$, on commencera par implémenter un simulateur dédié, en utilisant la méthode d'inversion.

1. Vérifiez que la fonction $g(x)$ correspond bien à une densité et représentez la graphiquement pour $x \in [-5, 5]$.
   — Remarque : pour les calculs, pensez à traiter séparément les cas où x est positif ou négatif pour faire dispararaître la valeur absolue.
2. Implémentez une procédure d'inversion permettant de simuler un échantillon distribué selon la densité $g(x)$. Vous détaillerez les calculs et le raisonnement suivi.
3. Validez votre procédure en simulant un échantillon de taille 1000 et en comparant graphiquement la densité obtenue empiriquement à la densité théorique.
4. Pour implémenter la procédure de rejet, il nous faut trouver un majorant M tel que $f(x) \leq Mg(x)$, $\forall x$ .Vérifier graphiquement que $M = \sqrt{\frac{2e}{\pi}}$ est le plus petit majorant. Pour un point bonus, démontrez le par le calcul.
5. Implémentez la procédure de rejet pour simuler un échantillon selon la densité $f(x)$. Validez votre procédure en simulant un échantillon de taille 1000 et en comparant avec la densité théorique de la loi normale centrée réduite. Quel est le taux de rejet de votre procédure ? Commentez.

2 Exercice 2 - approche Monte Carlo pour le dimensionnement d'un jeu de validation

A l'issue d'une procédure de validation croisée, on a estimé que le taux de bonne classification d'un classifieur vaut 95%. On souhaite à présent confirmer ces performances en appliquant le classifieur sur un échantillon de validation. L'acquisition de nouvelles données étant coûteuse,

on cherche à trouver la plus petite taille d'échantillon permettant de démontrer ce niveau de performance.

Implémenter une procédure de Monte Carlo permettant de quantifier la puissance de détecter l'hypothèse alternative :

$$H_1 : \text{le taux de bonne classification du modèle vaut } 0.95$$

par rapport aux hypothèses nulles :

$$H_0 : \text{le taux de bonne classification du modèle vaut } p_0,$$

pour $p_0 \in \{0.8, 0.85, 0.9, 0.92, 0.93\}$, en fonction de la taille d'échantillon, et pour un risque de première espèce $\alpha = 0.05$.

On considèrera des tailles d'échantillons comprises entre $n = 50$ et $n = 500$. On présentera les résultats sous la forme d'un graphique représentant la puissance en fonction de n pour les différentes hypothèses nulles considérées, et on commentera les résultats obtenus.

On note que le taux de bonne classification est une proportion et, que sous l'hypothèse nulle, on peut utiliser l'approximation suivante :

$$\frac{p - p_0}{\sqrt{\frac{p_0(1-p_0)}{n}}} \to \mathcal{N}(0, 1),$$

où p est la proportion mesurée dans l'échantillon.

3 Exercice 3 - comparaison de courbes ROC

On souhaite comparer les performances de deux classifieurs probabilistes du type régression logistique sur un jeu de données de validation. Charger pour cela le fichier `exo3.Rdata`. Il contient les données suivantes :

- un vecteur `y` contenant les catégories de référence, codées en -1 et $+1$, des 1639 observations.
- un vecteur `prob1` contenant les probabilités d'être de la catégorie $+1$ obtenues par le premier classifieur.
- un vecteur `prob2` contenant ces mêmes probabilités, obtenues par le second classifieur.

On cherche à comparer les performances de ces classifieurs selon le critère de l'aire sous la courbe ROC (AUC).

1. Rappeler brièvement le principe d'une courbe ROC. Comment peut-on interpréter l'AUC en terme probabiliste ?
2. Représenter les courbes ROC et calculer les AUC correspondantes [1]. Commenter l'allure des courbes ROC.
3. Calculer par boostrap un intervalle de confiance à 95% de ces AUC par la méthode des quantiles. Commenter les résultats obtenus et illustrer graphiquement la variabilité mesurée par bootstrap.
4. Proposer une procédure de permutation pour tester l'hypothèse que les AUC sont différentes. Comparer la p-valeur obtenue à celle produite par la fonction `roc.test` du package `pROC` par la méthode bootstrap, et commentez les résultats obtenus.

1. Vous pourrez par exemple utiliser le package `ROCR` utilisé l'an dernier dans l'UE Apprentissage Statistique I.

4 Exercice 4 - k plus proches voisins & bagging

On souhaite comparer les performances de l'algorithme des k plus proches voisins (k-ppv) et de sa version "bootstrappée" par l'algorithme du "bagging". Charger pour cela le jeu de données stocké dans le fichier `exo-4.Rdata`. Il contient des données d'apprentissage et de test permettant respectivement de construire et d'évaluer un classifieur :

— `X.train` et `X.test` sont les matrices de prédicteurs (on dispose de deux variables),
— `y.train` et `y.test` sont les catégories associées ("0" ou "1").

1. Représenter le jeu de données en faisant figurer sur une même figure les données de test et d'apprentissage. Utiliser des couleurs et/ou symboles permettant de différencier les deux catégories ainsi que les jeux d'apprentissage et de test.
2. Mesurer le taux de bonne classification obtenu par l'algorithme des k-ppv pour $k \in \{1, 3, 5, 7, 9, 11\}$.
3. Comparer ces résultats avec les performances obtenues par bagging en considérant $B = 100$ tirages.
4. Proposer une représentation graphique permettant de comparer les performances obtenues avec et sans bagging, et de visualiser les performances des classifieurs individuels aggrégés par bagging.

On rappelle qu'on peut appliquer l'algorithme des k-ppv via la fonction `kpp` du package `class` ainsi `> preds = knn(train, test, cl, k)` où :

— `train` et `test` sont respectivement les données d'apprentissage et de test,
— `cl` contient les catégories des données d'apprentissage,
— `k` spécifie le nombre de voisins à considérer.

Pour le bagging à proprement parler, vous implémenterez une procédure par vos propres moyens (i.e., sans utiliser d'éventuel package permettant d'automatiser la procédure).