Méthodes d'analyse biostatistique projet 2 (date limite : le 21 novembre)

Wen, Zehai; Li, Qingyue

2023-11-11 14:21:45-05:00

Exercice 1

Décrivez brièvement les variables du jeu de données `diabetic` dans la bibliothèque `survival` en R. Quelle était une question de recherche menant à cette collecte de données? Produisez un graphique de Kaplan-Meier utilisant l'échantillon entier, le groupe de `trt = 0` et le groupe de `trt = 1` pour le temps jusqu'à devenir aveugle. Pour le groupe `trt = 0`, trouvez le temps médian de survie et construisez un intervalle de confiance à 95% pour le temps médian de survie. Faites un test de Log-Rang pour comparer les deux groupes.

Finalement, on se demande si d'autres variables dans le jeu de données pourraient être des facteurs confondants, et si l'on devrait stratifier le test du Log-Rang sur l'une de ces variables. À partir de statistiques descriptives, de graphiques et/ou d'arguments adaptés au contexte de l'étude, discuter de laquelle des variables du jeu de données risque d'agir comme facteur confondant, et reproduire le test du Log-Rank stratifié pour cette variable.

Solution Tout d'abord, nous importons l'ensemble de données et examinons les informations de base sur les données.

```
library(rsq)

data("hcrabs")
attach (hcrabs)
```

Ensuite,nous réalisons un modèle de régression de Poisson avec la variable de réponse `num.satellites` et les variables explicatives `color`, `spine`, et `width`.

```
modele_poisson <-
glm ( num.satellites ~ width + spine + color, family = poisson ( link = log ) ,
data = hcrabs )
```

On fait puis le test de surdispersion et évaluer le paramèter de sur-dispersion.

```
library (AER)
print(dispersiontest(modele_poisson))
```

Le test donne que le paramètre de dispersion est 3.143975 avec p-valeur $4.07e-08$. Par conséquent, la sur-dispersion est significative. Alors la dispersion observée dans les données dépasse ce qui serait attendu sous une distribution de Poisson. Dans notre cas, les variations du nombre de satellites mâles ne sont pas bien prises en compte dans le modèle de Poisson standard et des modèles de surdiscrétisation plus complexes peuvent être envisagés pour mieux s'adapter à ces données. Lorsque nous effectuons une régression de Poisson, nous supposons que la variance est égale à la moyenne, une propriété inhérente à la distribution de Poisson. Cependant, les données réelles peuvent ne pas toujours respecter cette supposition.

Comme on a une surdispersion, il faudrait avoir une hétérogénéité non observée ou une accumulation de comptages issus de plusieurs processus indépendants. Face à la surdispersion, on se tourne généralement vers d'autres modèles plus complexes pour les données de comptage. Une choix est d'utilier un modèle mixte, qui s'appelle modèle binomial négatif. Il s'agit de commencer avec un modèle de Poisson avec le paramètre estimé par une distribution de Gamma.

Passons au sujet prochain, il y a $2 \times 4 - 1 = 15$ modèles possibles si on ne compte pas l'intercept. On choisit le critère AIC pour sélectionner le meilleur modèle. On rappelle que, si on a k modèles qui ont les valeurs d'AIC AIC_1, $\cdots$, AIC_k respectivement, les valeurs $\exp\{-\frac{1}{2}(\text{AIC}_i - \text{AIC}_{min})\}$ sont les probabilités que le modèle i minimisant la perte d'informations.

```
modele_color <- glm ( num.satellites ~ color, family = poisson ( link = log )
   ,data = hcrabs )
```

```
modele_spine <- glm ( num.satellites ~ spine, family = poisson ( link = log )
   ,data = hcrabs )
modele_width <- glm ( num.satellites ~ width, family = poisson ( link = log )
   ,data = hcrabs )
modele_weight <- glm ( num.satellites ~ weight, family = poisson ( link = log )
   ,data = hcrabs )
modele_color_spine <- glm ( num.satellites ~ color + spine, family = poisson (
   link = log ) ,data = hcrabs )
modele_color_width <- glm ( num.satellites ~ color + width, family = poisson (
   link = log ) ,data = hcrabs )
modele_color_weight <- glm ( num.satellites ~ color + weight, family = poisson (
   link = log ) ,data = hcrabs )
modele_spine_width <- glm ( num.satellites ~ spine + width, family = poisson (
   link = log ) ,data = hcrabs )
modele_spine_weight <- glm ( num.satellites ~ spine + weight, family = poisson (
   link = log ) ,data = hcrabs )
modele_width_weight <- glm ( num.satellites ~ width + weight, family = poisson (
   link = log ) ,data = hcrabs )
modele_color_spine_width <- glm ( num.satellites ~ color + spine + width, family
   = poisson ( link = log ) ,data = hcrabs )
modele_color_spine_weight <- glm ( num.satellites ~ color + spine + weight,
   family = poisson ( link = log ) ,data = hcrabs )
modele_color_width_weight <- glm ( num.satellites ~ color + width + weight,
   family = poisson ( link = log ) ,data = hcrabs )
modele_spine_width_weight <- glm ( num.satellites ~ spine + width + weight,
   family = poisson ( link = log ) ,data = hcrabs )
modele_color_spine_width_weight <- glm ( num.satellites ~ color + spine + width +
   weight, family = poisson ( link = log ) ,data = hcrabs )

modeles <- list(modele_color, modele_spine, modele_width, modele_weight,
modele_color_spine, modele_color_width, modele_color_weight, modele_spine_width,
modele_spine_weight, modele_width_weight, modele_color_spine_width,
modele_color_spine_weight, modele_color_width_weight,
modele_spine_width_weight, modele_color_spine_width_weight)

AICs <- rep(0, 15)
for (i in 1:15) {
  AICs[i] <- AIC(modeles[[i]])
}

print(AICs)

min_AIC <- min(AICs)
proba <- rep(0, 15)
for (i in 1:15) {
  proba[i] <- exp(0.5*(min_AIC- AICs[i]))
}

print(proba)

print(which.max(proba))
```

À la fin, on trouve que le modèle color-weight est le meilleur modèle. Pour le résidus, on choisit les

résidus d'Anscombe. Les résidus d'Anscombe sont spécifiquement conçus pour les modèles linéaires généralisés et offrent de bonnes propriétés pour identifier des valeurs atypiques ou des observations influentes. Ce lien `https://www.sfu.ca/sasdoc/sashtml/insight/chap39/sect57.htm` fournit des informations supplémentaires sur les résidus d'Anscombe.

```
library(surveillance)
plot(anscombe.residuals(modele_color_weight, phi =1))
```

Toutes les modélisations sont complètes.////

Exercice 2ab

Pour le jeu de données `rotterdam` dans la bibliothèque `survival` en R, on s'intéresse par l'association entre les variables explicatives `age`, `meno`, `sizes`, `grade`, `notes` et `chemo`, et la variable de réponse le temps jusqu'à une récidive du cancer au sein. Ajustez un modèle de Cox pour cela et rapporter les rapports de risque pour toutes les variables explicatives avec leurs intervalles de confiance (à 95% ?). Nommez l'hypothèse importante du modèle de Cox. Expliquez sans le faire comment on peut vérifier cette hypothèse.

Solution Tout d'abord, nous importons l'ensemble de données et examinons les informations de base sur les données.

```
library(rsq)

data("hcrabs")
attach (hcrabs)
```

Ensuite,nous réalisons un modèle de régression de Poisson avec la variable de réponse `num.satellites` et les variables explicatives `color`, `spine`, et `width`.

```
modele_poisson <-
glm ( num.satellites ~ width + spine + color, family = poisson ( link = log ) ,
data = hcrabs )
```

On fait puis le test de surdispersion et évaluer le paramèter de sur-dispersion.

```
library (AER)
print(dispersiontest(modele_poisson))
```

Le test donne que le paramètre de dispersion est 3.143975 avec p-valeur $4.07e-08$. Par conséquent, la sur-dispersion est significative. Alors la dispersion observée dans les données dépasse ce qui serait attendu sous une distribution de Poisson. Dans notre cas, les variations du nombre de satellites mâles ne sont pas bien prises en compte dans le modèle de Poisson standard et des modèles de surdiscrétisation plus complexes peuvent être envisagés pour mieux s'adapter à ces données. Lorsque nous effectuons une régression de Poisson, nous supposons que la variance est égale à la moyenne, une propriété inhérente à la distribution de Poisson. Cependant, les données réelles peuvent ne pas toujours respecter cette supposition.

Comme on a une surdispersion, il faudrait avoir une hétérogénéité non observée ou une accumulation de comptages issus de plusieurs processus indépendants. Face à la surdispersion, on se tourne généralement vers d'autres modèles plus complexes pour les données de comptage. Une choix est d'utilier un modèle mixte, qui s'appelle modèle binomial négatif. Il s'agit de commencer avec un modèle de Poisson avec le paramètre estimé par une distribution de Gamma.

Passons au sujet prochain, il y a $2 \times 4 - 1 = 15$ modèles possibles si on ne compte pas l'intercept. On choisit le critère AIC pour sélectionner le meilleur modèle. On rappelle que, si on a k modèles qui ont les valeurs d'AIC

$\text{AIC}_1, \cdots, \text{AIC}_k$ respectivement, les valeurs $\exp\{-\frac{1}{2}(\text{AIC}_i - \text{AIC}_{min})\}$ sont les probabilités que le modèle i minimisant la perte d'informations.

```
modele_color <- glm ( num.satellites ~ color, family = poisson ( link = log )
   ,data = hcrabs )
modele_spine <- glm ( num.satellites ~ spine, family = poisson ( link = log )
   ,data = hcrabs )
modele_width <- glm ( num.satellites ~ width, family = poisson ( link = log )
   ,data = hcrabs )
modele_weight <- glm ( num.satellites ~ weight, family = poisson ( link = log )
   ,data = hcrabs )
modele_color_spine <- glm ( num.satellites ~ color + spine, family = poisson (
   link = log ) ,data = hcrabs )
modele_color_width <- glm ( num.satellites ~ color + width, family = poisson (
   link = log ) ,data = hcrabs )
modele_color_weight <- glm ( num.satellites ~ color + weight, family = poisson (
   link = log ) ,data = hcrabs )
modele_spine_width <- glm ( num.satellites ~ spine + width, family = poisson (
   link = log ) ,data = hcrabs )
modele_spine_weight <- glm ( num.satellites ~ spine + weight, family = poisson (
   link = log ) ,data = hcrabs )
modele_width_weight <- glm ( num.satellites ~ width + weight, family = poisson (
   link = log ) ,data = hcrabs )
modele_color_spine_width <- glm ( num.satellites ~ color + spine + width, family
   = poisson ( link = log ) ,data = hcrabs )
modele_color_spine_weight <- glm ( num.satellites ~ color + spine + weight,
   family = poisson ( link = log ) ,data = hcrabs )
modele_color_width_weight <- glm ( num.satellites ~ color + width + weight,
   family = poisson ( link = log ) ,data = hcrabs )
modele_spine_width_weight <- glm ( num.satellites ~ spine + width + weight,
   family = poisson ( link = log ) ,data = hcrabs )
modele_color_spine_width_weight <- glm ( num.satellites ~ color + spine + width +
   weight, family = poisson ( link = log ) ,data = hcrabs )

modeles <- list(modele_color, modele_spine, modele_width, modele_weight,
modele_color_spine, modele_color_width, modele_color_weight, modele_spine_width,
modele_spine_weight, modele_width_weight, modele_color_spine_width,
modele_color_spine_weight, modele_color_width_weight,
modele_spine_width_weight, modele_color_spine_width_weight)

AICs <- rep(0, 15)
for (i in 1:15) {
  AICs[i] <- AIC(modeles[[i]])
}

print(AICs)

min_AIC <- min(AICs)
proba <- rep(0, 15)
for (i in 1:15) {
  proba[i] <- exp(0.5*(min_AIC- AICs[i]))
}
```

```
print(proba)

print(which.max(proba))
```

À la fin, on trouve que le modèle color-weight est le meilleur modèle. Pour le résidus, on choisit les résidus d'Anscombe. Les résidus d'Anscombe sont spécifiquement conçus pour les modèles linéaires généralisés et offrent de bonnes propriétés pour identifier des valeurs atypiques ou des observations influentes. Ce lien `https://www.sfu.ca/sasdoc/sashtml/insight/chap39/sect57.htm` fournit des informations supplémentaires sur les résidus d'Anscombe.

```
library(surveillance)
plot(anscombe.residuals(modele_color_weight, phi =1))
```

Toutes les modélisations sont complètes.////

Exercice 2c

Décrivez dans vos mots les trois tests statistiques discutés en Sections 5.3.1, 5.3.2, 5.3.3 du livre "Applied Survival Analysis Using R" de Moore (2016) pour les coefficients dans le modèle de Cox. Comme dans la Section 5.3., dans un deuxième temps, discutez/comparez entre les trois tests leurs avantages et inconvénients. Cette réponse devrait consister en environ 8 à 12 lignes max au total.

Solution Écrire $h_i(t_j)$ pour le hasard de patient i au temps d'échec t_j. Si l'hypothèse de proportionnalité est vraie, on peut écrire $h_i(t_j) = h_0(t_j)\psi_i = h_0(t_j)e^{z_i\beta}$ où h_0 est le hasard de référence, $z_i \in \{0, 1\}$ selon si le patient i est dans le groupe de traitement ou le groupe de contrôle et β est le paramètre à estimer. La vraisemblance partielle est définie par $L(\beta) = \prod_{k=1}^{D} \frac{\psi_i}{\sum_{l\in R_k}\psi_l}$ où D est le nombre d'échecs et R_k est l'ensemble des patients à risque au temps t_k. On teste l'hypothèse nulle $H_0 : \beta = 0$. Fixer un niveau $\alpha \in (0, \frac{1}{2})$. Écrire $l = \ln(L)$, $S = l'$, $I = -l''$ et soit $\hat{\beta} \in \arg\max_\beta L(\beta)$. Le test de Wald rejette H_0 si $\left|\hat{\beta}\sqrt{I(\hat{\beta})}\right| > z_{\frac{\alpha}{2}}$. Le test du score rejette H_0 si $\left|\frac{S(0)}{\sqrt{I(0)}}\right| > z_{\frac{\alpha}{2}}$. Le test du ratio vraisemblance rejette H_0 si $\left|2(l(\hat{\beta}) - l(0))\right| > \chi^2_{\frac{\alpha}{2},1}$. Si on ne peut pas obtenir $\hat{\beta}$, il faut utiliser le test du score. Supposons à partir de maintenant qu'on a obtient $\hat{\beta}$. Le résultat du test du ratio vraisemblance a un avantage par rapport aux deux autres tests qu'il est invariant contre les transformations monotones. Si l'invariance n'est pas nécessaire, alors le test de Wald est plus commun et plus facile à calculer.////

Exercice 3

Pour le jeu de données `datasurv.txt`, NOTATION

Solution Parmi les quatres stuctures mentionnés en classe, on décide que la matrice de corrélation est celui de Toepliz. On ne choisit pas les trois autres parce que les poids d'un individu entre les points différents de temps ne sont pas indépendents ou interchangeable et que l'on croit pas les poids forment une chaîne de Markov. On choisit Toepliz parce qu'il y a une « délai » entre les observations et que nous n'avons pas trouvé une autre matrice de corrélation en ligne qui semble plus appropriée.

D'abord, on réalise les trois modèles linéaires :

1. Le modèle avec seulement une ordonnée à l'origine :

```
modeles_lineaires_aucune <- list()
residues_aucune <- rep(1,4)
for (i in 1:4){
   modele_i <- lm(donnee_separe[[i]]$weight ~ 1)
   modeles_lineaires_aucune <- append(modeles_lineaires_aucune, modele_i)
   residues_aucune[i] <- mean(residuals(modele_i))
}
```

2. Le modèle avec `age` et `sex` comme variables explicatives :

```
modeles_lineaires_age_sex <- list()
residues_age_sex <- rep(1,4)
for (i in 1:4){
   modele_i <- lm(donnee_separe[[i]]$weight ~ donnee_separe[[i]]$age +
      donnee_separe[[i]]$sex)
   modeles_lineaires_age_sex <- append(modeles_lineaires_age_sex, modele_i)
   residues_age_sex[i] <- mean(residuals(modele_i))
}
```

3. Le modèle avec `age`, `sex`, `SES` et `smoking` comme variables explicatives :

```
modeles_lineaires_age_SES_sex_smoking <- list()
residues_age_SES_sex_smoking <- rep(1,4)
for (i in 1:4){
   modele_i <- lm(donnee_separe[[i]]$weight ~ donnee_separe[[i]]$age +
      donnee_separe[[i]]$sex +
donnee_separe[[i]]$SES + donnee_separe[[i]]$smoking)
   modeles_lineaires_age_SES_sex_smoking <- append(
modeles_lineaires_age_SES_sex_smoking, modele_i)
   residues_age_SES_sex_smoking[i] <- mean(residuals(modele_i))
}
```

On imprime puis les corrélations résiduelles :

```
print("Les correlations residuelles sont :")
print(cor(residues_aucune, residues_age_sex)) # -0.5644551
print(cor(residues_age_sex, residues_age_SES_sex_smoking)) # -0.9982123
print(cor(residues_age_SES_sex_smoking, residues_aucune)) # 0.5159088
```

- -0.5644551: Cette valeur représente la corrélation entre les résidus du premier modèle et ceux du deuxième modèle (avec *age* et *sex* comme variables explicatives). Cette corrélation négative signifie que lorsque les résidus du premier modèle augmentent, ceux du deuxième modèle ont tendance à diminuer, et vice versa.
- -0.9982123: Cette valeur représente la corrélation entre les résidus du deuxième modèle et ceux du troisième modèle. Cette corrélation, très proche de -1, indique une forte relation négative entre les résidus des deux modèles. Cela signifie que lorsque les résidus du deuxième modèle augmentent, ceux du troisième modèle ont tendance à diminuer, et vice versa.
- 0.5159088: Cette valeur représente la corrélation entre les résidus du troisième modèle et ceux du premier modèle. Cette corrélation positive signifie que lorsque les résidus du troisième modèle augmentent, ceux du premier modèle ont également tendance à augmenter, et vice versa.

À partir de maintenant, on utilise `SES` comme une variable explicative pour la variable de réponse `weight`. Les modèles démandés sont réalisés ci-dessous :

1. Modèle linéaire en assumant l'indépendance entre toutes les observations:

```
modele_lineaire <- lm(weight ~ SES, data = donnee)
print(summary(modele_lineaire))
```

   On trouve que l'écat type d'erreur pour `SES` est 0.1328.

2. Modèle linéaire généralisé (normale) où on assume une matrice de corrélation interchangeable:

```
library(nlme)
glm_normal <- gls(weight ~ SES, data = donnee, correlation =
   corCompSymm(form = ~ 1 | SES))
print(summary(glm_normal))
```

   On trouve que l'écat type d'erreur pour `SES` est 0.1868304.

3. GEE avec une matrice de corrélation interchangeable:

```
require(geepack)
GEE <- summary(geese(weight ~ SES, id = ID, data = donnee, corstr =
   'exchangeable'))
print(GEE)
```

   On trouve que l'écat type d'erreur pour `SES` est 0.1646921.

4. Modèle mixte avec une ordonnée à l'origine et une pente aléatoire pour chaque individu:

```
library(lme4)
modele_mixte <- lmer(weight ~ (1 + SES | ID), data = donnee)
print(summary(modele_mixte))
print(confint(modele_lineaire))
```

   Nous avons manipulé beaucoup d'options. Ce modèle ne converge jamais.

On sait que des erreurs standard plus petites impliquent des estimations plus précises des paramètres du modèle, tandis que des erreurs standard plus grandes indiquent une plus grande incertitude. Ainsi, parmi les quatre modèles mentionnés ci-dessus, le permier modèle présente l'erreur standard la plus faible (0.1328). Par conséquent, le premier modèle est le meilleur modèle pour les données. On puis construire le 95% intervalle de confiance :

```
print(confint(modele_lineaire))
```

L'intervalle pour l'ordonnée à l'origine est $[161.6002042, 165.2151458]$ et l'intervalle pour la pente est $[-0.6829711, -$
Donc, en général, on peut dire que le poids diminue lorsque `SES` augmente. ////