Modèles linéaires généralisés et données discrètes

Plan de cours

Anouar El Ghouch

LSBA, Université catholique de Louvain, Belgium

Le présent document fournit des informations clés à propos du cours et la procédure d'évaluation.

À la fin de ce document, se trouve une liste des packages R à charger, ainsi que le code de deux fonctions que nous utiliserons plus tard.

Contenu

Ce cours est divisé en quatre chapitres intitulés comme suit.

- Tableaux de contingence et tests Chi—2.
- Introduction aux modèles linéaires généralisés : Méthodologie et concepts généraux.
- La régression de Poisson.
- La régression logistique.

Méthodes d'apprentissage

Il s'agit d'une formation "classique", dont voici quelques aspects.

- Ce cours est axé sur les applications et comporte des éléments méthodologiques qui synthétisent les principes de base.
- Les transparents utilisés durant le cours en constituent le support principal.
- De nombreux exemples, avec des données réelles, seront utilisés pour illustrer les différents concepts.
- Le traitement et l'analyse des données sont effectués à l'aide de R. La connaissance de ce dernier est un pré-requis.
- R n'est qu'un outil de calcul et non un objectif d'apprentissage en soi.
- Tout le code R nécessaire pour produire les analyses présentées est fourni.
- La théorie du maximum de vraisemblance est également un pré-requis pour aborder ce cours avec aisance.
- Quatre sessions d'exercices sur ordinateur sont prévues.
- Ces séances mettront en pratique les différentes notions structurées durant le cours sur des cas bien ciblés.

Évaluation

- Un examen final écrit à livre ouvert sur 20. Il aura lieu en salle informatique, sauf en cas de problème technique majeur.
- L'examen sera composé de questions ouvertes et/ou QCM. Il aura une petite partie théorique (exemple: trouver à la main un estimateur de maximum de vraisemblance) et une grande partie sur ordinateur qui consiste à répondre à un questionnaire en exploitant les résultats d'analyses effectuées sur ordinateur.
- Les analyses à effectuer sont similaires à celles vues dans le cours et dans les TP's.
- On vous fournira les transparents du cours en version électronique (PDF). Vous ne devez apprendre aucune formule ou commande R par cœur.
- L'accent sera mis sur la compréhension des concepts et non pas sur la mémorisation des formules ou la manipulation de R.

Quelques informations pratiques

- Inscrivez-vous sur le site Moodle du cours. Vous y trouverez:
  - Slides présentés lors de cours.
  - Énoncés et solutions des TP's
  - Documents supplémentaires (jeux de données, complément d'information, ...).
- Livres de référence non obligatoires:
  - Categorical Data Analysis
  - Analysis of categorical data with R
  - Generalized Linear Models With Examples in R
- Toute remarque ou question est la bienvenue.

Packages R

Les packages suivants (voir l'objet `pkg` défini ci-dessous) seront utilisés ultérieurement. Pour les installer et les charger, il suffit d'exécuter le code ci-après.

```
pkg <- c("tidyverse",        # ggplots and some light data processing
         "vcd",              # visualization, summary and inference for contingency table
         "marginaleffects", # GLM predictions and visualization
         "statmod",          # calculation of randomized quantile residuals
         "glmulti",          # for best subset selection (exhaustive search)
         "faraway",          # to get the "gala" dataset used in an example
         "GGally",           # to get the ggpairs() function used for matrix plot
         "caret",            # predictive quality assessment, discriminatory power, cross-validation
         "mosaicData",       # to get the "Whickham" dataset used in an example
         "pROC"              # display and analyze ROC Curves
         )

Require <- function(...) {
  if (!require(..., quietly = TRUE, character.only = TRUE)) install.packages(..., quiet = TRUE)
  require(..., quietly = TRUE, character.only = TRUE)
}

pkg |> lapply(FUN = Require)
```

Fonctions R

Voici le code de deux fonctions qui seront utilisées ultérieurement.

1- La fonction pr2.R pour calculer le pseudo$-R^2$:

```
# ... : the name of one or more glm models for which Psudo-R2 is to be calculated
pr2 <- function(...) {
  list(...) |> sapply(\(glm) (1 - glm$deviance / glm$null.deviance) * 100)
}
```

2- Le fonction diagnost.R pour réaliser des graphiques de diagnostique:

<voir le Slide suivant>

```r
if (!require(statmod)) install.packages("statmod")

# mod: the glm model to diagnose
#
# type: the type of residuals to be calculated. This can be
#   #"rqr", for "randomized quantile residuals" (default). These are calculated using statmod::qresiduals()
#   #"deviance", or "pearson", or any other type of residuals from residuals()
#
# plots: Which plots to show? This can be
#   #"response", for reponses ~ fitteds
#   #"fitted", for residuals ~ fitteds
#   #"qqplot", to make a QQ-plot
#   #"formula", residuals ~ term, for all "terms" (covariates) appearing in the model formula
#   #"data", residuals ~ var, for all numerical covariates appearing in the provided data
#   #FALSE, return the residuals, do not plot
#   #a character vector of the names of the variables (in the provided data) against which the residuals are to be plotted

diagnost <- function(mod, type = "rqr", plots = c("response", "fitted", "qqplot", "formula", "data")) {
  if (type == "rqr") {
    prsd <- statmod::qresiduals(mod) |> pnorm()
    prsd[prsd == 1] <- 1 - 1e-010
    prsd[prsd == 0] <- 1e-010
    rsd <- prsd |> qnorm()
  } else {
    rsd <- residuals(mod, type = type)
  }
  if (length(plots) == 1) {
    if (plots == FALSE) {
      return(rsd)
    }
  }
  if (length(dev.list()) == 0) dev.new()
  opar <- par(no.readonly = TRUE)
  par(mar = c(3, 3, 1, 1), mgp = c(1.5, 0.5, 0), pch = 20)
  if ("formula" %in% plots) plots <- c(plots, attr(terms(mod), "term.labels")) |> unique()

  NumVarNames <- colnames(Filter(is.numeric, mod$data))
  resops <- all.vars(formula(mod))[1]
  if (resops %in% NumVarNames) NumVarNames <- NumVarNames[-which(NumVarNames == resops)]
  if ("data" %in% plots) plots <- c(plots, NumVarNames) |> unique()

  if (length(plots) > 1) {
    par(mfrow = c(1, 2))
  } else {
    par(mfrow = c(1, 1))
  }
  for (i in 1:length(plots)) {
    if (plots[i] == "response") {
      scatter.smooth(fitted(mod), mod$y, lpars = list(col = gray(0.6)), main = NULL, xlab = "Predicted Values", ylab = "Response")
      abline(a = 0, b = 1, col = "blue")
      mtext(side = 3, line = 0, adj = 0, cex = 0.8, paste("Response vs Fitted"), font = 2)
    } else if (plots[i] == "fitted") {
      scatter.smooth(fitted(mod), rsd, degree = 2, lpars = list(col = gray(0.6)), main = NULL, xlab = "Predicted Values", ylab = "Residuals")
      abline(h = 0, col = "blue")
      mtext(side = 3, line = 0, adj = 0, cex = 0.8, paste(type, ":", "Residuals vs Fitted"), font = 2)
    } else if (plots[i] == "qqplot") {
      qqnorm(rsd, main = "")
      qqline(rsd, col = "blue")
      mtext(side = 3, line = 0, adj = 0, cex = 0.8, paste(type, ":", "QQ-plot"), font = 2)
    } else if (plots[i] %in% names(mod$model)) {
      scatter.smooth(mod$model[, plots[i]], rsd, degree = 2, lpars = list(col = gray(0.6)), xlab = plots[i], main = NULL, ylab = "Residuals")
      abline(h = 0, col = "blue")
      mtext(side = 3, line = 0, adj = 0, cex = 0.8, paste(type, ":", "Residuals vs", plots[i]), font = 2)
    } else if (plots[i] %in% names(mod$data)) {
      scatter.smooth(mod$data[, plots[i]], rsd, degree = 2, xlab = plots[i], main = NULL, ylab = "Residuals")
      abline(h = 0, col = "blue")
      mtext(side = 3, line = 0, adj = 0, cex = 0.8, paste(type, ":", "Residuals vs", plots[i]), font = 2)
    }
  }
  opar$mfrow <- c(1, 1)
  on.exit(par(opar))
}
```