Solution Exercice #1, Série 3

Francis Duval

11 février 2020

Pour les énoncés des exercices, cliquer sur ce lien: https://nbviewer.jupyter.org/github/nmeraihi/ACT6100/blob/master/exercices_3.ipynb

Activer les librairies utiles.

```
library(here)
library(tidyverse)
library(magrittr)
library(glue)
```

Lire la base de données `swmotorcycle.rda`.

```
load(here("0_data", "swmotorcycle.rda"))
```

Pour voir rapidement à quoi ressemble la base de données, on peut utiliser la fonction `glimpse`.

```
glimpse(swmotorcycle)
```

```
## Observations: 64,548
## Variables: 9
## $ OwnerAge    <int> 0, 4, 5, 5, 6, 9, 9, 9, 10, 10, 10, 11, 11, 12, 12...
## $ Gender      <chr> "Male", "Male", "Female", "Female", "Female", "Fem...
## $ Area        <chr> "Central parts of Sweden's three largest cities", ...
## $ RiskClass   <chr> "EV ratio 13-15", "EV ratio 20-24", "EV ratio 9-12...
## $ VehAge      <int> 12, 9, 18, 25, 26, 8, 6, 20, 16, 17, 21, 13, 22, 1...
## $ BonusClass  <chr> "BM1", "BM1", "BM1", "BM1", "BM1", "BM1", "BM1", "...
## $ Exposure    <dbl> 0.175342, 0.000000, 0.454795, 0.172603, 0.180822, ...
## $ ClaimNb     <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ ClaimAmount <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
```

Partie 1

Premièrement, on va regarder combien de valeurs manquantes ont chacune des variables `OwnerAge`, `Exposure` et `ClaimNb` (parce que pour ajuster un GLM, aucune valeur ne doit être manquante).

```
swmotorcycle %>%
  select(ClaimNb, Exposure, OwnerAge) %>%
  map(~ sum(is.na(.)))
```

```
## $ClaimNb
## [1] 0
##
## $Exposure
## [1] 0
##
## $OwnerAge
## [1] 0
```

Parfait, aucune valeur n'est manquante. Cependant, on peut remarquer que la variable `Exposure` prend quelquefois la valeur zéro.

```
sum(swmotorcycle$Exposure == 0)
```

```
## [1] 2074
```

On voit que pour 2074 lignes de la base de données, l'exposition est nulle. Ça ne fonctionnera pas dans un GLM. On va donc supprimer ces lignes (avec la fonction `dplyr::filter`), et en même temps on va se créer une nouvelle base de données appelée `data_glm`.

```
data_glm <- swmotorcycle %>%
  filter(Exposure != 0)
```

Si on regarde les valeurs que la variable `OwnerAge` peut prendre, on voit que plusieurs assurés ont moins de 16 ans.

```
table(swmotorcycle$OwnerAge)
```

```
##
##    0    4    5    6    9   10   11   12   13   14   15   16   17   18   19
##    1    1    2    1    3    3    2    3    5    6   16  121  295  426  471
##   20   21   22   23   24   25   26   27   28   29   30   31   32   33   34
##  519  714  972 1308 1526 1638 1639 1604 1529 1464 1393 1264 1208 1082  993
##   35   36   37   38   39   40   41   42   43   44   45   46   47   48   49
##  978  939  890  895  945 1223 1579 1947 2046 2046 2075 2096 2068 2026 1963
##   50   51   52   53   54   55   56   57   58   59   60   61   62   63   64
## 1956 1905 1866 1710 1572 1441 1285 1161 1040  970  856  729  646  562  499
##   65   66   67   68   69   70   71   72   73   74   75   76   77   78   79
##  436  389  295  254  205  158  114  103   99   67   67   55   46   41   27
##   80   81   82   83   84   85   86   87   91   92
##   19    6   12   14    8    5    2    1    1    1
```

Par exemple, 1 assuré a 0 an, 1 assuré a 4 ans, 2 assurés ont 5 ans, etc. On va supposer qu'un assuré ne peut avoir que 16 ans ou plus, donc on va supprimer les contrats avec des assurés de 15 ans ou moins (avec la fonction `dplyr::filter`).

```
data_glm %<>%
  filter(OwnerAge > 15)
```

On va aussi seulement garder les variables utiles pour la modélisation (avec la fonction `dplyr::select`).

```
data_glm %<>%
  select(ClaimNb, Exposure, OwnerAge)
```

On ajuste ensuite un GLM Poisson avec fonction de lien logarithmique et avec la variable `Exposure` en offset.

```
glm_pois_fit <- glm(
  ClaimNb ~ OwnerAge,
  family = poisson(link = "log"),
  offset = log(Exposure),
  data = data_glm
)

summary(glm_pois_fit)
```

```
##
## Call:
## glm(formula = ClaimNb ~ OwnerAge, family = poisson(link = "log"),
##     data = data_glm, offset = log(Exposure))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9648  -0.1641  -0.1147  -0.0785   5.1087
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.13447    0.11952  -17.86   <2e-16 ***
## OwnerAge    -0.06035    0.00319  -18.92   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 6647.6  on 62435  degrees of freedom
## Residual deviance: 6279.1  on 62434  degrees of freedom
## AIC: 7631.7
##
## Number of Fisher Scoring iterations: 7
```

Les coefficients sont

```
coef(glm_pois_fit)
```

```
## (Intercept)    OwnerAge
## -2.13446732 -0.06035487
```

Partie 2

Pour déterminer la valeur optimale de K sans validation croisée, on pourrait par exemple essayer toutes les valeurs possibles de K, calculer l'erreur quadratique moyenne (EQM) pour toutes ces valeurs et garder la valeur de K qui minimise l'EQM. Observons premièrement quelles valeurs la variable `OwnerAge` peut prendre, ainsi que le nombre de valeurs possibles.

```
unique(data_glm$OwnerAge)
```

```
##  [1] 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38
## [24] 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61
## [47] 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84
## [70] 85 86 87 92
```

```
length(unique(data_glm$OwnerAge))
```

```
## [1] 73
```

On voit que la variable `OwnerAge` peut prendre 73 valeurs différentes, soient 16, ..., 92. Il y a donc 72 manières de séparer la variable `OwnerAge` en 2 catégories (K = 16, ..., 91). Cependant, puisque la valeur `OwnerAge` ne prend pas souvent des valeurs en haut de 75, on n'essayera pas ces valeurs. On va donc se créer 59 bases de données (avec chacune une valeur de K différente), ajuster un GLM Poisson sur chacune, calculer l'EQM pour chaque modèle et finalement, choisir la valeur de K qui mène à la plus petite EQM.

On commence par créer les 72 bases de données, qu'on va stocker dans la liste `data_glm_ls`.

```
# Fonction qui ajoute une variable age catégorielle à 2 classes (spécifier valeur k)
creer_variable_age_2_classes <- function(k) {
  data_glm %>%
    mutate(
      OwnerAge_2_classes = factor(if_else(OwnerAge <= k, glue("{k} ans ou moins"), glue("Plus de {k} ans")))
    )
}

# Obtenir toutes les valeurs de k possibles
(valeurs_k <- head(unique(data_glm$OwnerAge), -14))
```

```
##  [1] 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38
## [24] 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61
## [47] 62 63 64 65 66 67 68 69 70 71 72 73 74
```

```
# Obtenir une liste avec les 59 bases de données
data_glm_ls <- map(valeurs_k, creer_variable_age_2_classes)
```

Observons par exemple la structure du 10eme élément de cette liste.

```
glimpse(data_glm_ls[[10]])
```

```
## Observations: 62,436
## Variables: 4
## $ ClaimNb            <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
```

```
## $ Exposure           <dbl> 0.232877, 0.501370, 0.336986, 0.249315, 0.4...
## $ OwnerAge           <int> 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16,...
## $ OwnerAge_2_classes <fct> 25 ans ou moins, 25 ans ou moins, 25 ans ou...
```

Ensuite on ajuste un GLM Poisson sur chacune des bases de données de cette liste, qu'on stocke aussi dans une liste.

```
glm_pois_fit_ls <- map(
  data_glm_ls,
  ~ glm(ClaimNb ~ OwnerAge_2_classes, family = poisson(link = "log"), offset = log(Exposure), data = .x)
)
```

Ensuite on calcule l'EQM pour chacun des modèles

```
(eqm_vec <- map_dbl(glm_pois_fit_ls, ~ mean((.x$fitted.values - .x$y) ^ 2)))
```

```
##  [1] 0.01189707 0.01189881 0.01189030 0.01188980 0.01188754 0.01188606
##  [7] 0.01187763 0.01187666 0.01186835 0.01185963 0.01185064 0.01184044
## [13] 0.01184004 0.01182343 0.01182061 0.01182150 0.01182345 0.01182187
## [19] 0.01182465 0.01182545 0.01182700 0.01182588 0.01182937 0.01182897
## [25] 0.01183312 0.01184315 0.01185236 0.01187941 0.01188351 0.01189620
## [31] 0.01191125 0.01191976 0.01192066 0.01191915 0.01189809 0.01190008
## [37] 0.01190106 0.01190270 0.01190283 0.01190452 0.01190337 0.01190312
## [43] 0.01190299 0.01190186 0.01190244 0.01190162 0.01190053 0.01190011
## [49] 0.01190016 0.01189948 0.01189905 0.01189877 0.01189921 0.01189886
## [55] 0.01189859 0.01189840 0.01189822 0.01189806 0.01189795
```

```
(k_opt_eqm <- valeurs_k[which.min(eqm_vec)])
```

```
## [1] 30
```

La valeur optimale est K = 30 selon l'EQM Voici le code pour obtenir la fréquence espérée de chacune des 2 classes de risque (30 ans ou moins et plus de 30 ans):

```
new_data <- tibble(
  Exposure = c(1, 1),
  OwnerAge_2_classes = factor(c("30 ans ou moins", "Plus de 30 ans"))
)

new_data %>%
  mutate(
    frequence_esperee = round(
      predict(glm_pois_fit_ls[[which.min(eqm_vec)]], newdata = ., type = "response"),
      digits = 4
    )
  )
```

```
## # A tibble: 2 x 3
##   Exposure OwnerAge_2_classes frequence_esperee
##      <dbl> <fct>                          <dbl>
## 1        1 30 ans ou moins               0.0313
## 2        1 Plus de 30 ans                0.0063
```

Maintenant, on va déterminer la valeur optimale de K, mais avec la validation croisée. On va premièrement créer une fonction qui prend en entrée une base de données et qui retourne l'EQM de validation croisée avec 12 "folds".

```
# Définir une fonction qui prend en entrée un nombre n et renvoie 12 "folds" de taille égale
create_folds <- function(n, theseed = 2020) {
  set.seed(theseed)
  seq(1, n) %>%
    cut(breaks = 12, labels = F) %>%
    sample() %>%
    factor()
}
```

```r
# Fonction qui prend en entrée une base de données et renvoie l'eqm de validation croisée
cv_12_folds_mse_glm_poisson <- function(data) {
  data %<>% mutate(fold = create_folds(nrow(.)))
  responses_ls <- data %>% group_split(fold) %>% map(~ pull(., ClaimNb))

  models_ls <- map(
    1:12,
    ~ glm(
        ClaimNb ~ OwnerAge_2_classes,
        family = poisson(link = "log"),
        offset = log(Exposure),
        data = filter(data, fold != .)
      )
  )

  predictions_ls <- map(
    1:12,
    ~ predict(models_ls[[.]], newdata = filter(data, fold == .), type = "response")
  )

  res <- map2(responses_ls, predictions_ls, ~ mean((.x - .y) ^ 2)) %>% flatten_dbl() %>% mean()
  return(res)
}
```

Ensuite on calcule l'erreur quadratique moyenne de validation croisée pour chacune des bases de données dans la liste `data_glm_ls`.

```r
(eqm_vec <- map_dbl(data_glm_ls, cv_12_folds_mse_glm_poisson))
```

```
##  [1] 0.01189829 0.01189989 0.01189397 0.01189190 0.01188968 0.01188791
##  [7] 0.01187883 0.01187746 0.01186943 0.01186051 0.01185220 0.01184214
## [13] 0.01184154 0.01182491 0.01182213 0.01182263 0.01182457 0.01182274
## [19] 0.01182566 0.01182658 0.01182803 0.01182689 0.01183037 0.01182991
## [25] 0.01183397 0.01184387 0.01185328 0.01188031 0.01188408 0.01189680
## [31] 0.01191177 0.01192057 0.01192135 0.01191994 0.01189832 0.01190062
## [37] 0.01190188 0.01190333 0.01190320 0.01190495 0.01190378 0.01190361
## [43] 0.01190346 0.01190224 0.01190281 0.01190204 0.01190096 0.01190051
## [49] 0.01190056 0.01189988 0.01189944 0.01189914 0.01189954 0.01189920
## [55] 0.01189893 0.01189873 0.01189856 0.01189840 0.01189829
```

La valeur de K optimale s'obtient avec le code suivant:

```r
(k_opt_eqm <- valeurs_k[which.min(eqm_vec)])
```

```
## [1] 30
```

On rajuste un GLM Poisson avec la valeur optimale de K sur la base de données complète et on calcule les fréquences espérées:

```r
glm_pois_fit_opt_k_eqm <- glm(
  ClaimNb ~ OwnerAge_2_classes,
  family = poisson(link = "log"),
  offset = log(Exposure),
  data = data_glm_ls[[which.min(eqm_vec)]]
)

new_data_eqm <- tibble(
  Exposure = c(1, 1),
  OwnerAge_2_classes = factor(c("30 ans ou moins", "Plus de 30 ans"))
)

new_data_eqm %>%
  mutate(
    frequence_esperee = round(predict(glm_pois_fit_opt_k_eqm, newdata = ., type = "response"), 4)
```

```
  )
```

```
## # A tibble: 2 x 3
##   Exposure OwnerAge_2_classes frequence_esperee
##      <dbl> <fct>                          <dbl>
## 1        1 30 ans ou moins               0.0313
## 2        1 Plus de 30 ans                0.0063
```