Modèle linéaire généralisé et choix de modèles

Devoir maison obligatoire (Dauphine)

Paul Hardouin

August 31, 2019

Contents

1. Introduction **2**

1.1 Données et objectif . . . 2

1.2 Type de modèle à trouver . . . 2

1.3 Quatre méthodes pour trouver les modèles . . . 2

2. Lecture et exploration des données **3**

2.1 Lecture . . . 3

2.2 Visualisation des données . . . 3

3. Ajout et suppression de covariables **5**

3.1 Ajout de covariables . . . 5

3.2 Suppression de covariables inutiles . . . 6

4. Construction des 4 modèles **7**

4.1 Modèle 1 : MANUEL : sélection de covariables en observant les données 7

4.2 Modèle 2 : stepAIC : recherche pas à pas en utilisant le critère AIC . . 9

4.3 Modèle 3 : stepBIC : recherche pas à pas en utilisant le critère BIC . . 11

4.4 Modèle 4 : COMPLET : utilisation de toutes les covariables . . . 13

5. ACP sur les modèles **16**

6. Etude de performance par validation croisée **18**

6.1 Fonction pour faire la validation croisée . . . 18

6.2 Application de la validation croisée avec 10 blocs . . . 19

6.3 Observation des résultats . . . 19

7. Application du modèle retenu au jeu de test **21**

1. Introduction

1.1 Données et objectif

Pour cette étude, nous travaillons sur un jeu de données [source : **MeteoBlue**] sur les conditions météorologiques à Bale. Chaque ligne correspond à un jour entre 2010 et 2018. On y trouve les mesures de paramètres divers, ainsi qu'un booléen **pluie.demain** informant de la présence de pluie le lendemain du jour concerné.

1.2 Type de modèle à trouver

L'objectif est de trouver une valeur booléenne en fonction de covariables à valeurs continues.
Il s'agit donc d'un problème de régréssion logistique.

Compte-tenu de la nature du problème, nous cherchons un modèle **LOGIT [glm, family = binomial]**
On ne s'intéresse pas au probit, car la probabilité recherchée n'est pas très proche de 0 ou de 1.

Nous renoncons aux modèles suivants

- **PROBIT** [glm, family = binomial(link = "probit")]
- **BINOMIAL** [glm + cbind, family = binomial]
- **MULTINOMIAL** [multinom]
- **CUMULATIF** [clm]
- **POISSON** [glm, family = poisson]

1.3 Quatre méthodes pour trouver les modèles

- **MANUEL** : sélection de covariables en observant les données
- **stepAIC** : recherche pas à pas en utilisant le critère AIC
- **stepBIC** : recherche pas à pas en utilisant le critère BIC
- **COMPLET** : utilisation de toutes les covariables

2. Lecture et exploration des données

2.1 Lecture

```
d = read.csv("meteo.train.csv")
#summary(d)
```

2.2 Visualisation des données

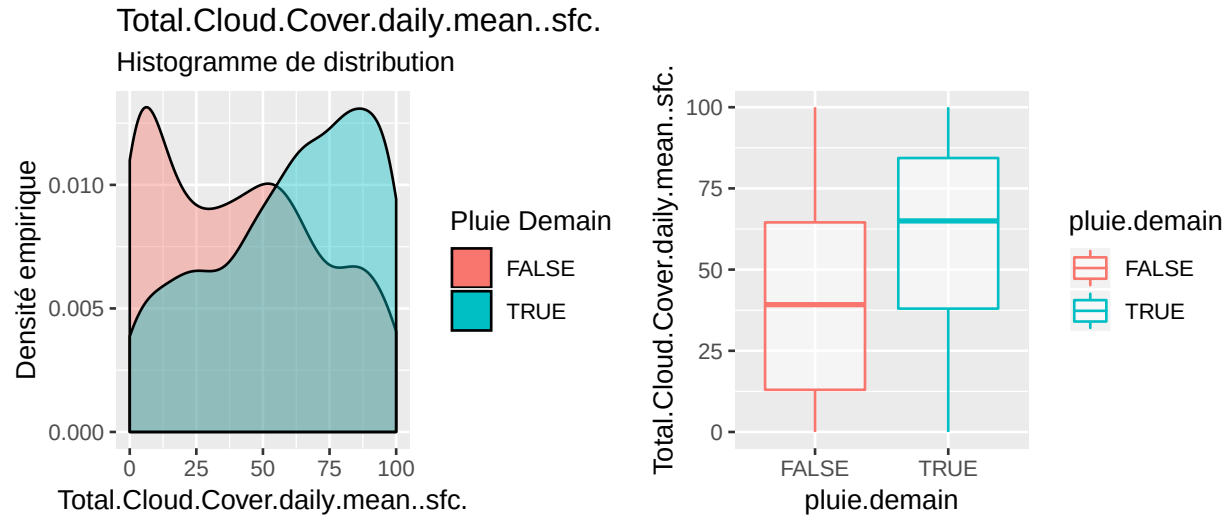
On veut pouvoir observer facilement l'influence des covariables sur **pluie.demain**
On crée donc la fonction **myVisu** pour cela

```
myVisu = function(d,ind,exportOption){
  # 0. Load library
    library(ggplot2)
    library(egg)
  # 1. Create histogramme
    p = ggplot(d, aes_string(x = colnames(d)[ind], fill = "pluie.demain"))
    hp = p + labs(title = colnames(d)[ind], x = colnames(d)[ind],  y = "Densité empirique",
                  fill = "Pluie Demain", subtitle = "Histogramme de distribution") +
             geom_density(alpha = 0.4) + # Transparency
             guides(fill = guide_legend(override.aes = list(alpha = 1)))
  # 2. Create a box plot (bp)
    p = ggplot(d, aes_string(x = "pluie.demain", y = colnames(d)[ind]))
    bp = p + geom_boxplot(aes_string(color = "pluie.demain"), alpha=0.5)
  # 3. Display
    figure = ggarrange(hp, bp, ncol = 2, nrow = 1)
  # 4. Export
    if (exportOption){ggsave(sprintf("meteo_feature_%02d.png",ind), plot = figure, width = 11, height
  # 5. Update indice
    return(figure)
}
```

```
# for(i in 1:ncol(d)){
#   if(length(unique(d[, i])) >= 2 & colnames(d)[i] != "pluie.demain")
#   {
#     toto = myVisu(d,i,1)
#   }
# }
```

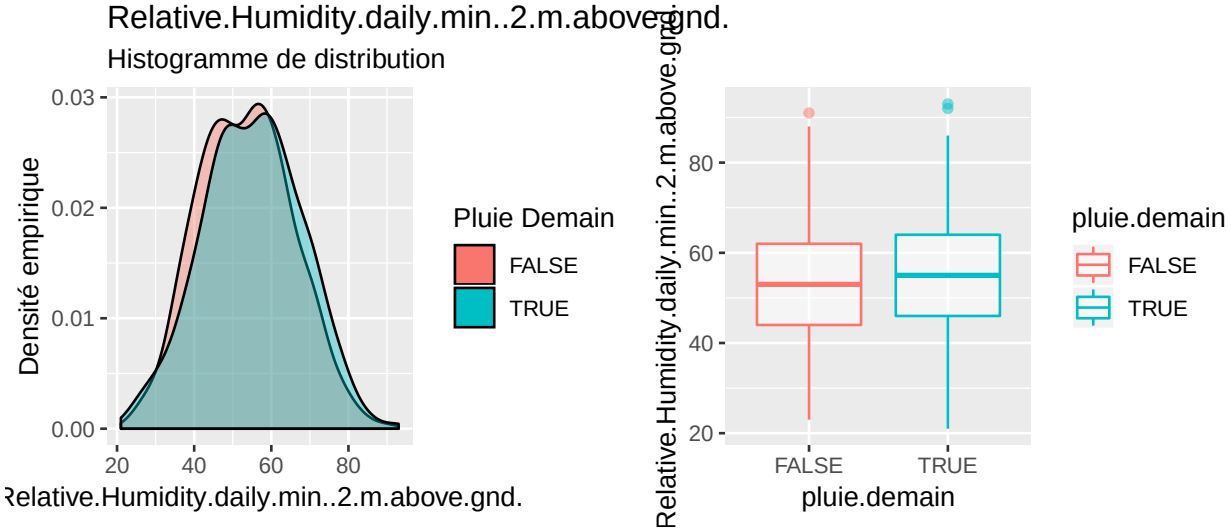
On observe des covariables qui semblent pouvoir influer **pluie.demain**
En effet, les distributions **TRUE/FALSE** sont différentes.

```
library(gridExtra)
#myFig=myVisu(d, 9,0)
myFig=myVisu(d,12,0)
```



A l'inverse, on voit aussi des covariables pour lesquelles les distributions **TRUE/FALSE** sont très proches.

```
#myFig=myVisu(d, 8,0)
myFig=myVisu(d,28,0)
```



3. Ajout et suppression de covariables

3.1 Ajout de covariables

On sent intuitivement que le cycle des saisons peut avoir une influence sur la pluie.
Pour appliquer une régréssion linéaire sur une telle information, on veut créer 2 covariables:

- le **cosinus** du **jour julien**
- le **sinus** du **jour julien**

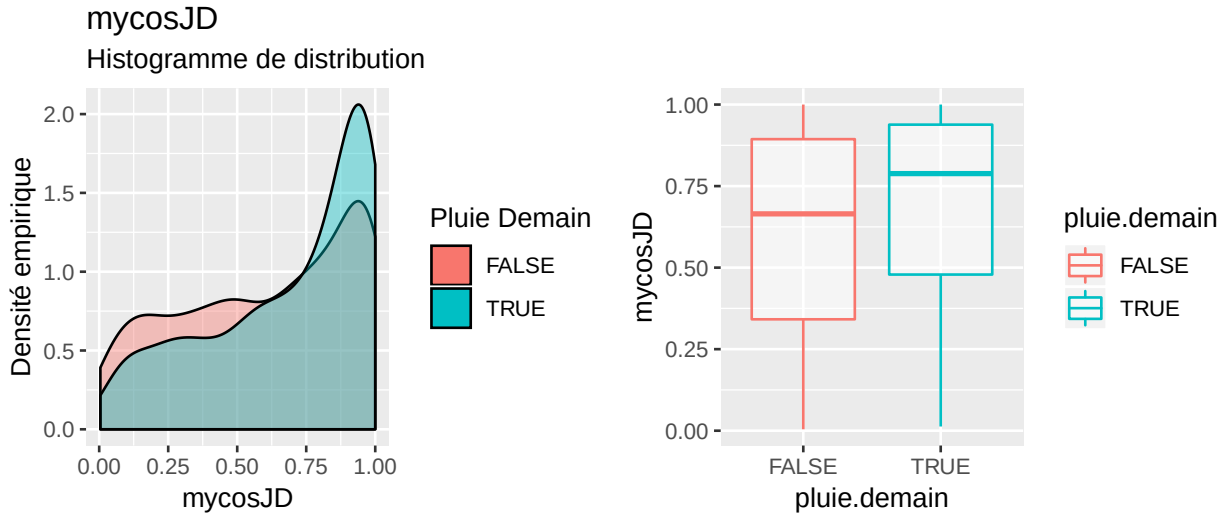
On s'apercoit en réalite que la vraie périodicité est plutot de 6 mois.
Les covariables créées sont donc les suivantes:

- la valeur absolue du **cosinus** du **jour julien**
- la valeur absolue du **sinus** du **jour julien**

```
tmp = do.call(paste, list(d$Month, d$Day, d$Year))
tmp = as.Date(tmp, format=c("%m %d %Y"))
d$mycosJD <- abs(cos(as.numeric(format(tmp, "%j"))/365*2*pi))
d$mysinJD <- abs(sin(as.numeric(format(tmp, "%j"))/365*2*pi))
```

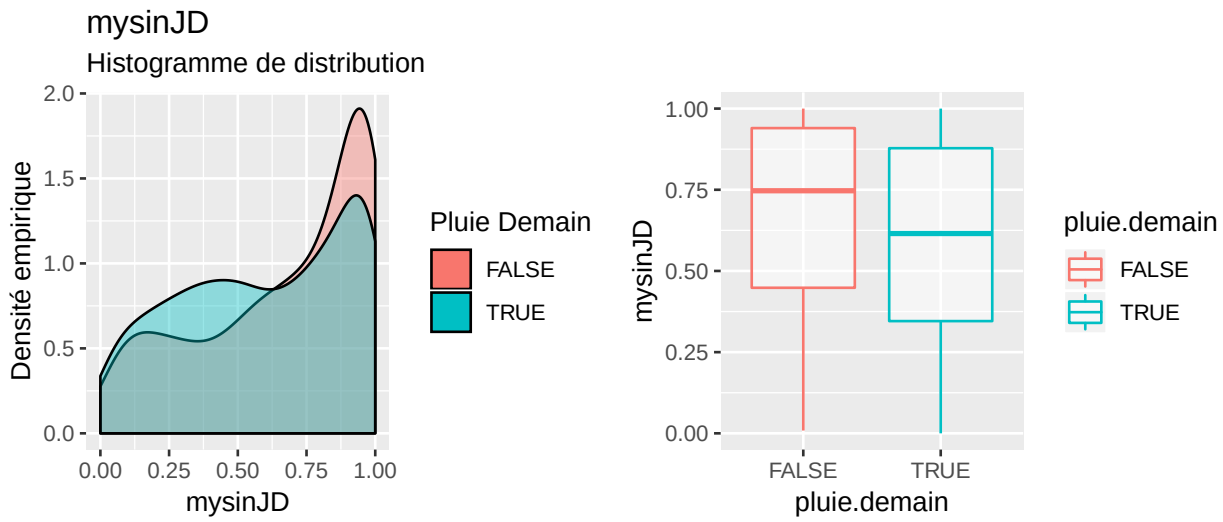
On visualise la distribution suivante pour **mycosJD**

```
myFig=myVisu(d,48,0)
```



On visualise la distribution suivante pour **mysinJD**

```
myFig=myVisu(d,49,0)
```



3.2 Suppression de covariables inutiles

On supprime quelques covariables encombrantes. Les **heures** et les **minutes**.

```
d = d[,!colnames(d) %in% c("Hour", "Minute")]
```

4. Construction des 4 modèles

4.1 Modèle 1 : MANUEL : sélection de covariables en observant les données

Pour ce modèle, on fait une selection manuelle des covariables. Pour cela, on exporte **myVisu** pour toutes les covariables vues en haut. On sélectionne alors manuellement les covariables aui nous intéressent.

On obtient un modèle avec 11 covariables.

```
m_perso = glm(formula = pluie.demain ~ Mean.Sea.Level.Pressure.daily.mean..MSL. +
                                       Mean.Sea.Level.Pressure.daily.max..MSL. +
                                       Mean.Sea.Level.Pressure.daily.min..MSL. +
                                       Total.Cloud.Cover.daily.mean..sfc. +
                                       Sunshine.Duration.daily.sum..sfc. +
                                       Wind.Direction.daily.mean..10.m.above.gnd. +
                                       Wind.Direction.daily.mean..80.m.above.gnd. +
                                       Wind.Direction.daily.mean..900.mb. +
                                       High.Cloud.Cover.daily.max..high.cld.lay. +
                                       mycosJD +
                                       mysinJD,
        family = binomial,
        data = d)
f_perso= formula(m_perso) # formule du modèle
n_perso = names(m_perso$coefficients) # nom des covariables retenues
summary(m_perso)
```

```
##
## Call:
## glm(formula = pluie.demain ~ Mean.Sea.Level.Pressure.daily.mean..MSL. +
##     Mean.Sea.Level.Pressure.daily.max..MSL. + Mean.Sea.Level.Pressure.daily.min..MSL. +
##     Total.Cloud.Cover.daily.mean..sfc. + Sunshine.Duration.daily.sum..sfc. +
##     Wind.Direction.daily.mean..10.m.above.gnd. + Wind.Direction.daily.mean..80.m.above.gnd. +
##     Wind.Direction.daily.mean..900.mb. + High.Cloud.Cover.daily.max..high.cld.lay. +
##     mycosJD + mysinJD, family = binomial, data = d)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2670  -0.9134  -0.2571   0.9338   2.3929
##
## Coefficients:
##                                          Estimate Std. Error z value
## (Intercept)                             98.9140668 10.9297545   9.050
## Mean.Sea.Level.Pressure.daily.mean..MSL.    0.1880782  0.1135358   1.657
## Mean.Sea.Level.Pressure.daily.max..MSL.    -0.1391341  0.0627025  -2.219
## Mean.Sea.Level.Pressure.daily.min..MSL.    -0.1477216  0.0594406  -2.485
## Total.Cloud.Cover.daily.mean..sfc.          0.0042020  0.0057882   0.726
## Sunshine.Duration.daily.sum..sfc.          -0.0004445  0.0005938  -0.749
## Wind.Direction.daily.mean..10.m.above.gnd.  0.0040049  0.0048134   0.832
## Wind.Direction.daily.mean..80.m.above.gnd. -0.0063310  0.0050283  -1.259
## Wind.Direction.daily.mean..900.mb.          0.0055116  0.0011953   4.611
```

```
## High.Cloud.Cover.daily.max..high.cld.lay.   0.0076935  0.0017157   4.484
## mycosJD                                     0.7765446  0.5408170   1.436
## mysinJD                                    -0.3002482  0.5397814  -0.556
##                                            Pr(>|z|)
## (Intercept)                                 < 2e-16 ***
## Mean.Sea.Level.Pressure.daily.mean..MSL.     0.0976 .
## Mean.Sea.Level.Pressure.daily.max..MSL.      0.0265 *
## Mean.Sea.Level.Pressure.daily.min..MSL.      0.0129 *
## Total.Cloud.Cover.daily.mean..sfc.           0.4679
## Sunshine.Duration.daily.sum..sfc.            0.4541
## Wind.Direction.daily.mean..10.m.above.gnd.   0.4054
## Wind.Direction.daily.mean..80.m.above.gnd.   0.2080
## Wind.Direction.daily.mean..900.mb.         4.01e-06 ***
## High.Cloud.Cover.daily.max..high.cld.lay.  7.32e-06 ***
## mycosJD                                      0.1510
## mysinJD                                      0.5780
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1724.5  on 1243  degrees of freedom
## Residual deviance: 1403.8  on 1232  degrees of freedom
## AIC: 1427.8
##
## Number of Fisher Scoring iterations: 4
```

4.2 Modèle 2 : stepAIC : recherche pas à pas en utilisant le critère AIC

Pour ce modèle, on fait une recherche pas à pas en utilisant le critère AIC (par defaut). On fait une recherche avec la méthode progressive.

```
fit1 = glm(pluie.demain ~ ., family = binomial, data = d)
fit2 = glm(pluie.demain ~ 1, family = binomial, data = d)
m_AIC = step(fit2,direction="both"    ,scope=list(upper=fit1,lower=fit2))
```

AIC : on obtient un modèle avec 9 covariables.

```
f_AIC = formula(m_AIC) # formule du modèle
n_AIC = names(m_AIC$coefficients) # nom des covariables retenues
summary(m_AIC)
```

```
##
## Call:
## glm(formula = pluie.demain ~ Mean.Sea.Level.Pressure.daily.min..MSL. +
##     Medium.Cloud.Cover.daily.max..mid.cld.lay. + mysinJD + Wind.Direction.daily.mean..900.mb. +
##     High.Cloud.Cover.daily.mean..high.cld.lay. + Mean.Sea.Level.Pressure.daily.max..MSL. +
##     Mean.Sea.Level.Pressure.daily.mean..MSL. + Total.Cloud.Cover.daily.max..sfc. +
##     Low.Cloud.Cover.daily.mean..low.cld.lay., family = binomial,
##     data = d)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1953  -0.9195  -0.2912   0.9264   2.3678
##
## Coefficients:
##                                          Estimate Std. Error z value
## (Intercept)                             92.9910621 10.5539094   8.811
## Mean.Sea.Level.Pressure.daily.min..MSL.    -0.1543399  0.0598128  -2.580
## Medium.Cloud.Cover.daily.max..mid.cld.lay.  0.0111587  0.0024315   4.589
## mysinJD                                 -0.9920768  0.2181659  -4.547
## Wind.Direction.daily.mean..900.mb.          0.0038405  0.0009557   4.019
## High.Cloud.Cover.daily.mean..high.cld.lay.  0.0104611  0.0036656   2.854
## Mean.Sea.Level.Pressure.daily.max..MSL.    -0.1444229  0.0620434  -2.328
## Mean.Sea.Level.Pressure.daily.mean..MSL.    0.2064663  0.1135267   1.819
## Total.Cloud.Cover.daily.max..sfc.          -0.0061293  0.0033261  -1.843
## Low.Cloud.Cover.daily.mean..low.cld.lay.    0.0038018  0.0024171   1.573
##                                         Pr(>|z|)
## (Intercept)                              < 2e-16 ***
## Mean.Sea.Level.Pressure.daily.min..MSL.     0.00987 **
## Medium.Cloud.Cover.daily.max..mid.cld.lay. 4.45e-06 ***
## mysinJD                                 5.43e-06 ***
## Wind.Direction.daily.mean..900.mb.          5.86e-05 ***
## High.Cloud.Cover.daily.mean..high.cld.lay.  0.00432 **
## Mean.Sea.Level.Pressure.daily.max..MSL.     0.01992 *
## Mean.Sea.Level.Pressure.daily.mean..MSL.    0.06896 .
## Total.Cloud.Cover.daily.max..sfc.           0.06536 .
## Low.Cloud.Cover.daily.mean..low.cld.lay.    0.11575
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1724.5  on 1243  degrees of freedom
## Residual deviance: 1394.8  on 1234  degrees of freedom
## AIC: 1414.8
##
## Number of Fisher Scoring iterations: 4
```

4.3 Modèle 3 : stepBIC : recherche pas à pas en utilisant le critère BIC

Pour ce modèle, on fait une recherche pas à pas en utilisant le critère BIC. Ce critère est plus favorable aux modèles restreint en nombre de covariables. Pour cela, on change la valeur du parametre **k** de la fonction **step**. Pour AIC, on a $\mathbf{k = 2}$. Pour BIC, on a $\mathbf{k = log(nrow(d))}$ On fait une recherche avec la méthode progressive.

```
fit1 = glm(pluie.demain ~ ., family = binomial, data = d)
fit2 = glm(pluie.demain ~ 1, family = binomial, data = d)
m_BIC = step(fit2,direction="both"    ,scope=list(upper=fit1,lower=fit2) ,k=log(nrow(d)))
```

BIC : on obtient un modèle avec 5 covariables.
Cela était prévisible, car BIC privilégie les modèles avec peu de covariables.

```
f_BIC = formula(m_BIC) # formule du modèle
n_BIC = names(m_BIC$coefficients) # nom des covariables retenues
summary(m_BIC)
```

```
##
## Call:
## glm(formula = pluie.demain ~ Mean.Sea.Level.Pressure.daily.min..MSL. +
##     Medium.Cloud.Cover.daily.max..mid.cld.lay. + mysinJD + Wind.Direction.daily.mean..900.mb. +
##     High.Cloud.Cover.daily.mean..high.cld.lay., family = binomial,
##     data = d)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0793  -0.9220  -0.3024   0.9385   2.3060
##
## Coefficients:
##                                          Estimate Std. Error z value
## (Intercept)                             87.2840206  9.7798757   8.925
## Mean.Sea.Level.Pressure.daily.min..MSL.  -0.0871380  0.0095828  -9.093
## Medium.Cloud.Cover.daily.max..mid.cld.lay.  0.0092403  0.0019891   4.646
## mysinJD                                 -0.9811577  0.2176748  -4.507
## Wind.Direction.daily.mean..900.mb.          0.0037996  0.0009394   4.045
## High.Cloud.Cover.daily.mean..high.cld.lay.  0.0104911  0.0036566   2.869
##                                          Pr(>|z|)
## (Intercept)                               < 2e-16 ***
## Mean.Sea.Level.Pressure.daily.min..MSL.   < 2e-16 ***
## Medium.Cloud.Cover.daily.max..mid.cld.lay. 3.39e-06 ***
## mysinJD                                  6.56e-06 ***
## Wind.Direction.daily.mean..900.mb.       5.24e-05 ***
## High.Cloud.Cover.daily.mean..high.cld.lay.  0.00412 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1724.5  on 1243  degrees of freedom
## Residual deviance: 1404.5  on 1238  degrees of freedom
## AIC: 1416.5
```

```
##
## Number of Fisher Scoring iterations: 4
```

4.4 Modèle 4 : COMPLET : utilisation de toutes les covariables

Pour ce modèle, on prend toutes les covariables du jeu de données.

```
m_full = glm(formula = pluie.demain ~ .,
        family = binomial,
        data = d)
print(mean(abs(round(predict(m_full, d, type = "response"))-d$pluie.demain)))
```

```
## [1] 0.278135
```

```
f_full = formula(m_full) # formule du modèle
n_full = colnames(d) # nom des covariables retenues
summary(m_full)
```

```
##
## Call:
## glm(formula = pluie.demain ~ ., family = binomial, data = d)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7240  -0.8609  -0.2049   0.8833   2.3841
##
## Coefficients:
##                                              Estimate Std. Error z value
## (Intercept)                                -6.653e+04  9.189e+04  -0.724
## X                                          -9.059e-02  1.251e-01  -0.724
## Year                                        3.314e+01  4.571e+01   0.725
## Month                                       2.767e+00  3.806e+00   0.727
## Day                                         8.086e-02  1.250e-01   0.647
## Temperature.daily.mean..2.m.above.gnd.      8.910e-02  1.643e-01   0.542
## Relative.Humidity.daily.mean..2.m.above.gnd. -4.739e-02  3.016e-02  -1.571
## Mean.Sea.Level.Pressure.daily.mean..MSL.    1.557e-01  1.279e-01   1.218
## Total.Precipitation.daily.sum..sfc.         1.100e-02  2.433e-02   0.452
## Snowfall.amount.raw.daily.sum..sfc.         2.499e-01  3.103e-01   0.805
## Total.Cloud.Cover.daily.mean..sfc.         -4.240e-03  1.210e-02  -0.351
## High.Cloud.Cover.daily.mean..high.cld.lay.  5.503e-03  6.827e-03   0.806
## Medium.Cloud.Cover.daily.mean..mid.cld.lay. -2.699e-03  6.457e-03  -0.418
## Low.Cloud.Cover.daily.mean..low.cld.lay.    9.919e-03  8.248e-03   1.203
## Sunshine.Duration.daily.sum..sfc.          -6.663e-04  8.806e-04  -0.757
## Shortwave.Radiation.daily.sum..sfc.        -2.216e-05  9.990e-05  -0.222
## Wind.Speed.daily.mean..10.m.above.gnd.     -2.851e-02  9.462e-02  -0.301
## Wind.Direction.daily.mean..10.m.above.gnd.  9.916e-03  5.190e-03   1.911
## Wind.Speed.daily.mean..80.m.above.gnd.     -1.124e-01  6.589e-02  -1.706
## Wind.Direction.daily.mean..80.m.above.gnd. -1.077e-02  5.367e-03  -2.006
## Wind.Speed.daily.mean..900.mb.              1.328e-02  2.307e-02   0.576
## Wind.Direction.daily.mean..900.mb.          6.938e-03  1.394e-03   4.977
## Wind.Gust.daily.mean..sfc.                  3.783e-02  3.570e-02   1.059
## Temperature.daily.max..2.m.above.gnd.       1.122e-01  9.650e-02   1.163
## Temperature.daily.min..2.m.above.gnd.      -2.277e-01  8.586e-02  -2.652
## Relative.Humidity.daily.max..2.m.above.gnd.  2.859e-02  1.958e-02   1.460
## Relative.Humidity.daily.min..2.m.above.gnd.  2.561e-02  1.663e-02   1.540
## Mean.Sea.Level.Pressure.daily.max..MSL.    -1.454e-01  7.074e-02  -2.056
```

```
## Mean.Sea.Level.Pressure.daily.min..MSL.      -1.063e-01  6.706e-02  -1.584
## Total.Cloud.Cover.daily.max..sfc.            -4.559e-03  4.787e-03  -0.952
## Total.Cloud.Cover.daily.min..sfc.             1.798e-02  8.042e-03   2.236
## High.Cloud.Cover.daily.max..high.cld.lay.     2.678e-03  2.785e-03   0.962
## High.Cloud.Cover.daily.min..high.cld.lay.    -2.667e-02  2.569e-02  -1.038
## Medium.Cloud.Cover.daily.max..mid.cld.lay.    1.049e-02  3.120e-03   3.361
## Medium.Cloud.Cover.daily.min..mid.cld.lay.    4.047e-03  9.944e-03   0.407
## Low.Cloud.Cover.daily.max..low.cld.lay.      -8.774e-04  3.381e-03  -0.260
## Low.Cloud.Cover.daily.min..low.cld.lay.      -1.818e-02  8.067e-03  -2.254
## Wind.Speed.daily.max..10.m.above.gnd.         2.086e-02  3.369e-02   0.619
## Wind.Speed.daily.min..10.m.above.gnd.         1.093e-01  5.850e-02   1.868
## Wind.Speed.daily.max..80.m.above.gnd.        -8.994e-04  2.764e-02  -0.033
## Wind.Speed.daily.min..80.m.above.gnd.         8.916e-03  4.076e-02   0.219
## Wind.Speed.daily.max..900.mb.                 4.287e-03  1.085e-02   0.395
## Wind.Speed.daily.min..900.mb.                -1.549e-02  1.725e-02  -0.898
## Wind.Gust.daily.max..sfc.                     7.163e-03  1.735e-02   0.413
## Wind.Gust.daily.min..sfc.                    -2.135e-02  2.646e-02  -0.807
## mycosJD                                       1.064e+00  5.840e-01   1.822
## mysinJD                                      -2.838e-01  5.777e-01  -0.491
##                                              Pr(>|z|)
## (Intercept)                                  0.469083
## X                                            0.469067
## Year                                         0.468459
## Month                                        0.467280
## Day                                          0.517654
## Temperature.daily.mean..2.m.above.gnd.       0.587541
## Relative.Humidity.daily.mean..2.m.above.gnd. 0.116095
## Mean.Sea.Level.Pressure.daily.mean..MSL.     0.223327
## Total.Precipitation.daily.sum..sfc.          0.651031
## Snowfall.amount.raw.daily.sum..sfc.          0.420565
## Total.Cloud.Cover.daily.mean..sfc.           0.725927
## High.Cloud.Cover.daily.mean..high.cld.lay.   0.420148
## Medium.Cloud.Cover.daily.mean..mid.cld.lay.  0.675904
## Low.Cloud.Cover.daily.mean..low.cld.lay.     0.229161
## Sunshine.Duration.daily.sum..sfc.            0.449267
## Shortwave.Radiation.daily.sum..sfc.          0.824469
## Wind.Speed.daily.mean..10.m.above.gnd.       0.763181
## Wind.Direction.daily.mean..10.m.above.gnd.   0.056067 .
## Wind.Speed.daily.mean..80.m.above.gnd.       0.087973 .
## Wind.Direction.daily.mean..80.m.above.gnd.   0.044809 *
## Wind.Speed.daily.mean..900.mb.               0.564750
## Wind.Direction.daily.mean..900.mb.           6.45e-07 ***
## Wind.Gust.daily.mean..sfc.                   0.289415
## Temperature.daily.max..2.m.above.gnd.        0.244970
## Temperature.daily.min..2.m.above.gnd.        0.008013 **
## Relative.Humidity.daily.max..2.m.above.gnd.  0.144385
## Relative.Humidity.daily.min..2.m.above.gnd.  0.123450
## Mean.Sea.Level.Pressure.daily.max..MSL.      0.039801 *
## Mean.Sea.Level.Pressure.daily.min..MSL.      0.113091
## Total.Cloud.Cover.daily.max..sfc.            0.340882
## Total.Cloud.Cover.daily.min..sfc.            0.025364 *
## High.Cloud.Cover.daily.max..high.cld.lay.    0.336152
## High.Cloud.Cover.daily.min..high.cld.lay.    0.299270
## Medium.Cloud.Cover.daily.max..mid.cld.lay.   0.000777 ***
```

```
## Medium.Cloud.Cover.daily.min..mid.cld.lay.   0.684032
## Low.Cloud.Cover.daily.max..low.cld.lay.      0.795244
## Low.Cloud.Cover.daily.min..low.cld.lay.      0.024209 *
## Wind.Speed.daily.max..10.m.above.gnd.        0.535736
## Wind.Speed.daily.min..10.m.above.gnd.        0.061714 .
## Wind.Speed.daily.max..80.m.above.gnd.        0.974046
## Wind.Speed.daily.min..80.m.above.gnd.        0.826857
## Wind.Speed.daily.max..900.mb.                0.692902
## Wind.Speed.daily.min..900.mb.                0.369347
## Wind.Gust.daily.max..sfc.                    0.679679
## Wind.Gust.daily.min..sfc.                    0.419794
## mycosJD                                      0.068403 .
## mysinJD                                      0.623208
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1724.5  on 1243  degrees of freedom
## Residual deviance: 1325.8  on 1197  degrees of freedom
## AIC: 1419.8
##
## Number of Fisher Scoring iterations: 5
```

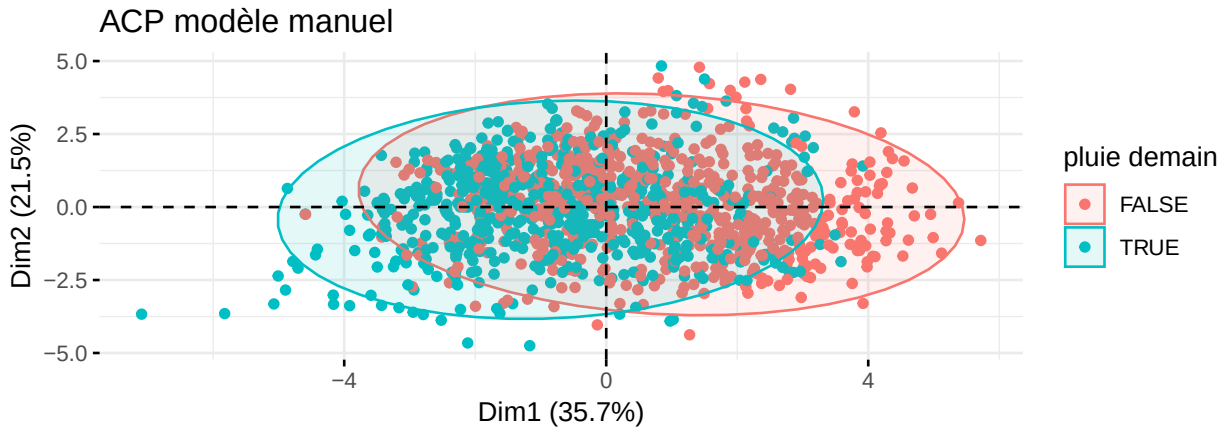
5. ACP sur les modèles

On affiche les ACP des différents modèles, pour voir si certains discriminent mieux **pluie.demain**

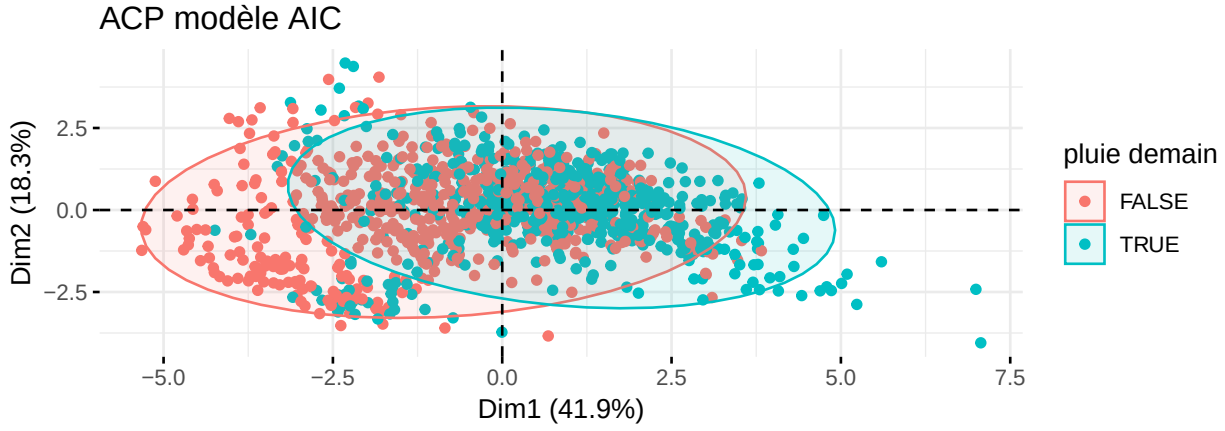
- Le modèle BIC semble le mieux *(61% expliqués par 2 PC + ellipses un peu séparées)*
- Ensuite, les modèle AIC *(60% expliqués par 2 PC + ellipses un peu séparées)*
- Ensuite, le modèle manuel *(58% expliqués par 2 PC + ellipses un peu séparées)*
- Enfin, le modèle complet semble plus médiocre *(40% expliqués par 2 PC + ellipses très proches)*

```
library('FactoMineR')
library("factoextra")
```

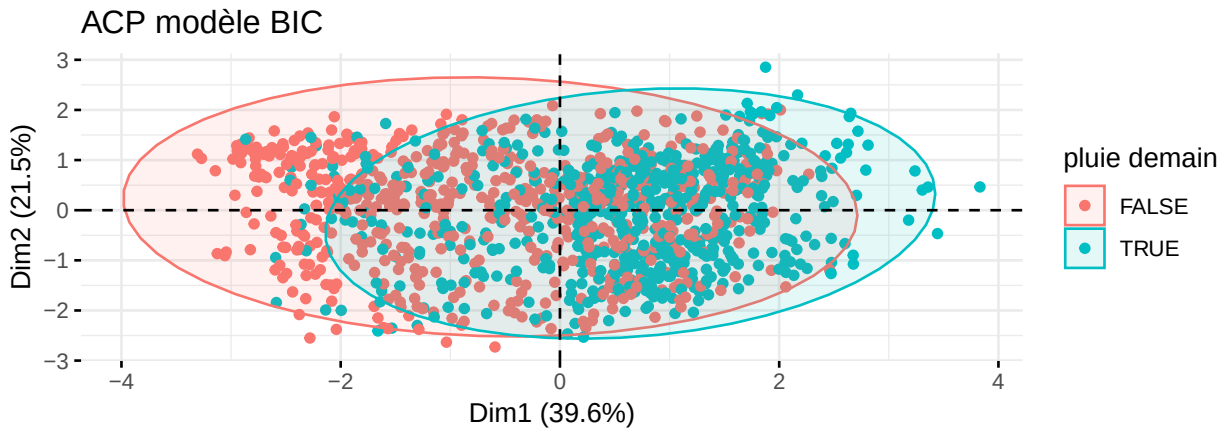
```
D = d[,(colnames(d) %in% n_perso) | colnames(d)=="pluie.demain"]
res.pca = PCA(D,quali.sup = which(colnames(D)=="pluie.demain"),graph=FALSE)
fviz_pca_ind (res.pca, geom.ind="point", col.ind=d$pluie.demain ,
              legend.title="pluie demain", addEllipses = T, title="ACP modèle manuel")
```



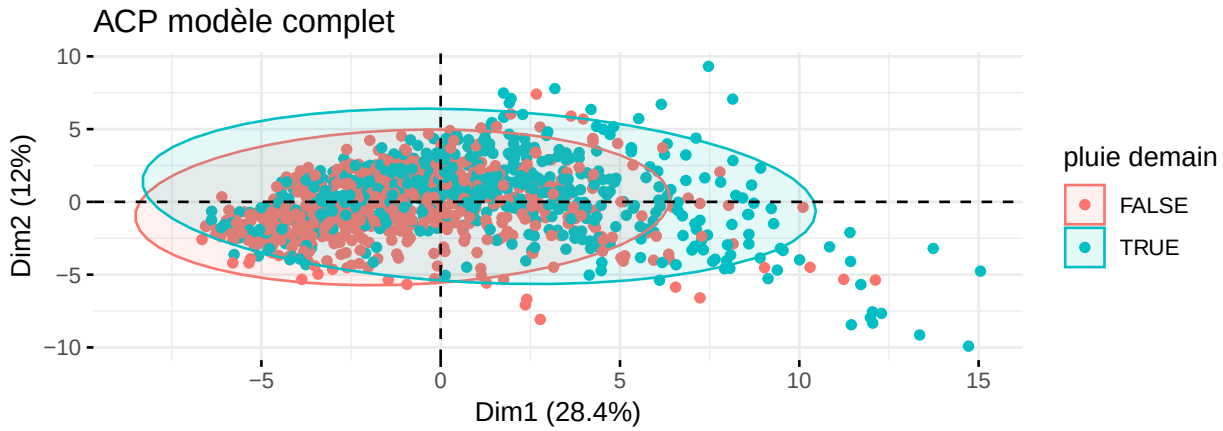

```
D = d[,(colnames(d) %in% n_AIC) | colnames(d)=="pluie.demain"]
res.pca = PCA(D,quali.sup = which(colnames(D)=="pluie.demain"),graph=FALSE)
fviz_pca_ind (res.pca, geom.ind="point", col.ind=d$pluie.demain ,
              legend.title="pluie demain", addEllipses = T, title="ACP modèle AIC")
```



```
D = d[,(colnames(d) %in% n_BIC) | colnames(d)=="pluie.demain"]
res.pca = PCA(D,quali.sup = which(colnames(D)=="pluie.demain"),graph=FALSE)
fviz_pca_ind (res.pca, geom.ind="point", col.ind=d$pluie.demain ,
              legend.title="pluie demain", addEllipses = T, title="ACP modèle BIC")
```




```
D = d
res.pca = PCA(D,quali.sup = which(colnames(D)=="pluie.demain"),graph=FALSE)
fviz_pca_ind (res.pca, geom.ind="point", col.ind=d$pluie.demain ,
              legend.title="pluie demain", addEllipses = T, title="ACP modèle complet")
```




```
toto = data.frame(res.pca$ind$coord,res.pca$call$quali.sup$quali.sup)
toto$pluie.demain = toto$pluie.demain=="TRUE"
```

6. Etude de performance par validation croisée

6.1 Fonction pour faire la validation croisée

Métrique : Pour ce calcul, on cherche a mesurer le taux d'erreur de prédiciction. On compare donc les vraies valeur de **pluie.demain** au arrondis des prédictions.

Partitions : Comme les données consécutives ont de la cohérence, on ne va pas les couper en K blocs consécutifs. A la place, pour le kieme bloc, on prend une ligne toutes les K lignes, en commencant par la kieme ligne.

Critère de choix : MOYENNE des erreurs : On choisira le modèle donnant le moins d'erreur moyenne sur la cross-validation. Les coefficients finaux seront donc la moyenne ponderée des coefficients de la validation croisèe. Le modèle contenant ces coefficient finaux est la seconde sortie de la fonction.

```
myCrossValidation = function(formule,dataFrame,nParts)
{
  # Initialization
    errV = numeric(0)
  # Boucle for sur le nombre de parties
    for (k in 1:nParts)
    {
      # Calcul des indices du jeu de test
        indTest = seq(k,nrow(dataFrame),nParts)
        df_test = dataFrame[indTest,]
        df_train = dataFrame[-indTest,]
      # Calcul du modèle
        modele = glm(formule, family = binomial, data = df_train)
      # Calcul des coefficient du modèle final
        if (k==1)
          { MODEL = modele
            N_TRAIN = nrow(df_train)
            MODEL$coefficients = modele$coefficients*N_TRAIN
            N_SUM = N_TRAIN }
        else
          { N_TRAIN = nrow(df_train)
            MODEL$coefficients = MODEL$coefficients + modele$coefficients*N_TRAIN
            N_SUM = N_SUM + N_TRAIN }
      # Prédiction pour la validation crois
        pred = predict(modele, df_test, type = "response")
      # Calcul de l'erreur de prediction
        err = mean(abs(round(pred)-df_test$pluie.demain))
        #print(mean(abs(round(predict(modele, dataFrame, type = "response"))-dataFrame$pluie.demain)))
      # Ajout au vecteur
        errV = rbind(errV,err)
    }
  # Output
    MODEL$coefficients = MODEL$coefficients/N_SUM
    return(list(erreur = errV, modele = MODEL))
}
```

6.2 Application de la validation croisée avec 10 blocs

```
cv_perso = myCrossValidation(f_perso, d, 10)
cv_AIC   = myCrossValidation(f_AIC  , d, 10)
cv_BIC   = myCrossValidation(f_BIC  , d, 10)
cv_full  = myCrossValidation(pluie.demain ~ . , d, 10)
cv_acp   = myCrossValidation(pluie.demain ~ . , toto, 10)
```

6.3 Observation des résultats

Voici l'ensemble des taux d'erreur pour les différentes méthodes

```
myResults = data.frame(cv_perso$erreur,cv_AIC$erreur,cv_BIC$erreur,cv_full$erreur,cv_acp$erreur)
myResults
```

```
##       cv_perso.erreur cv_AIC.erreur cv_BIC.erreur cv_full.erreur
## err         0.3840000     0.3760000     0.3760000      0.3200000
## err.1       0.2960000     0.3280000     0.3120000      0.3040000
## err.2       0.3040000     0.3040000     0.2800000      0.3200000
## err.3       0.3760000     0.3440000     0.3680000      0.3600000
## err.4       0.3145161     0.3548387     0.3145161      0.3548387
## err.5       0.2903226     0.2903226     0.2580645      0.2419355
## err.6       0.2903226     0.2822581     0.2903226      0.2822581
## err.7       0.3225806     0.3306452     0.3225806      0.3387097
## err.8       0.2338710     0.2419355     0.2419355      0.2822581
## err.9       0.2500000     0.2661290     0.2419355      0.2580645
##       cv_acp.erreur
## err       0.3920000
## err.1     0.3120000
## err.2     0.2400000
## err.3     0.3760000
## err.4     0.3387097
## err.5     0.2741935
## err.6     0.2903226
## err.7     0.3064516
## err.8     0.2500000
## err.9     0.2741935
```

En observant les métriques ci-dessous, on choisit de retenir **le modèle BIC**. En effet, le **le modèle BIC** obtient la moyenne la plus faible avec **0.3005**

```
summary(myResults)
```

```
##  cv_perso.erreur  cv_AIC.erreur    cv_BIC.erreur    cv_full.erreur
##  Min.   :0.2339   Min.   :0.2419   Min.   :0.2419   Min.   :0.2419
##  1st Qu.:0.2903   1st Qu.:0.2843   1st Qu.:0.2635   1st Qu.:0.2823
##  Median :0.3000   Median :0.3160   Median :0.3012   Median :0.3120
##  Mean   :0.3062   Mean   :0.3118   Mean   :0.3005   Mean   :0.3062
```

```
##  3rd Qu.:0.3206   3rd Qu.:0.3407   3rd Qu.:0.3206   3rd Qu.:0.3340
##  Max.   :0.3840   Max.   :0.3760   Max.   :0.3760   Max.   :0.3600
##  cv_acp.erreur
##  Min.   :0.2400
##  1st Qu.:0.2742
##  Median :0.2984
##  Mean   :0.3054
##  3rd Qu.:0.3320
##  Max.   :0.3920
```

7. Application du modèle retenu au jeu de test

- Lecture des données + création des features **mycosJD** et **mysinJD**.
- Prédiction avec le modèle moyenné **BIC** retenu.
- Nettoyage des features.
- Export du data frame dans le fichier **hardouin_meteo.predict.csv**.

```
# Lecture des données à prédire
  df_test = read.csv("meteo.test.csv")
# Ajout de mycosJD et mysinJD
  tmp = do.call(paste, list(df_test$Month, df_test$Day, df_test$Year))
  tmp = as.Date(tmp, format=c("%m %d %Y"))
  df_test$mycosJD <- abs(cos(as.numeric(format(tmp, "%j"))/365*2*pi))
  df_test$mysinJD <- abs(sin(as.numeric(format(tmp, "%j"))/365*2*pi))
# Prédiction
  prediction = round(predict(cv_BIC$modele, df_test, type = "response"))
# Nettoyage
  df_test = df_test[,c("X","Year","Month","Day")]
  df_test$pluie.demain.prediction = prediction
# Export
  write.csv(df_test,"hardouin_meteo.predict.csv")
```