Projet final : Analyse de données et apprentissage statistique

Challenge data de l'ENS, Plume Labs, "Predict air quality at the street level"

Laura Misrachi

Dimanche 21 janvier 2018

Contents

Ce compte rendu de projet d'analyse de données porte sur un jeu de données issu d'une start-up parisienne Plume Labs qui travaille sur la prévision de la qualité de l'air et développe un capteur connecté de la qualité de l'air. Le **but** du challenge est de prédire la concentration de trois polluants ($NO_2, PM_{10}, PM_{2.5}$) à niveau de rue pour différentes villes françaises, à différents instants (heures). Les données dont nous disposons et que nous allons examiner de façon approfondie par la suite s'apparentent donc à une série temporelle.

1 Importation des données

On importe ici les données issues de l'ensemble d'apprentissage et de test, qui sont fournis sur le site *"www.challengedata.ens.fr"*.

2 Présentation des données

```
## [1] "Covariables pour l'ensemble d'apprentissage"
```

```
##  [1] "ID"                "hlres_50"          "green_5000"
##  [4] "hldres_50"         "daytime"           "route_100"
##  [7] "precipintensity"   "precipprobability" "hlres_1000"
## [10] "temperature"       "is_calmday"        "route_1000"
## [13] "roadinvdist"       "port_5000"         "windbearingsin"
## [16] "cloudcover"        "hldres_100"        "natural_5000"
## [19] "hlres_300"         "hldres_300"        "route_300"
## [22] "station_id"        "pressure"          "route_500"
## [25] "hlres_500"         "hlres_100"         "pollutant"
## [28] "industry_1000"     "zone_id"           "windbearingcos"
## [31] "windspeed"         "hldres_500"        "hldres_1000"
```

```
## [1] "Covariables pour l'ensemble de test"
```

```
##  [1] "ID"                "hlres_50"          "green_5000"
##  [4] "hldres_50"         "daytime"           "route_100"
##  [7] "precipintensity"   "precipprobability" "hlres_1000"
## [10] "temperature"       "is_calmday"        "route_1000"
## [13] "roadinvdist"       "port_5000"         "windbearingsin"
## [16] "cloudcover"        "hldres_100"        "natural_5000"
## [19] "hlres_300"         "hldres_300"        "route_300"
## [22] "station_id"        "pressure"          "route_500"
## [25] "hlres_500"         "hlres_100"         "pollutant"
## [28] "industry_1000"     "zone_id"           "windbearingcos"
## [31] "windspeed"         "hldres_500"        "hldres_1000"
```

```
## [1] "Covariables pour le fichier réponse à soumettre pour le challenge"
```

```
## [1] "ID"     "TARGET"
```

Dès maintenant, il est important de préciser que l'ensemble de test fourni par l'entreprise rassemble des données pour lesquelles nous ne connaissons absolument pas les quantités exactes à prédire. C'est un ensemble utilisé pour estimer les performances de notre modèle par les hôtes du challenge et s'assurer que les participants ne fasse pas de sur-apprentissage sur l'ensemble à tester. Il nous faudra donc par ailleurs définir un autre ensemble de test pour lequel nous pouvons effectivement tester nous-même la qualité de notre modèle.

D'après les observations précédentes, les deux datasets d'apprentissage et de test contiennent respectivement 33 covariables identiques. Il est intéressant de noter que chacun des deux ensembles contiennent les mêmes covariables. Aucune modification ne devra être faite à ce niveau-là pour faire correspondre les deux datasets. Nous allons désormais davantage investiguer la nature et la structure des données.

```
## [1] "Structure des données pour l'ensemble d'apprentissage"
```

```
## 'data.frame':    448169 obs. of  33 variables:
##  $ ID               : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ hlres_50         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ green_5000       : num  5172542 5172542 5172542 5172542 5172542 ...
##  $ hldres_50        : num  3755 3755 3755 3755 3755 ...
##  $ daytime          : num  72 72 73 73 74 74 75 75 76 76 ...
##  $ route_100        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ precipintensity  : num  0.6096 0.6096 0.0965 0.0965 0 ...
##  $ precipprobability: num  0.61 0.61 0.14 0.14 0 ...
##  $ hlres_1000       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ temperature      : num  9.49 9.49 8.22 8.22 7.58 ...
```

```
##  $ is_calmday       : Factor w/ 2 levels "False","True": 1 1 1 1 1 1 1 1 1 1 ...
##  $ route_1000       : num  8027 8027 8027 8027 8027 ...
##  $ roadinvdist      : num  0.0468 0.0468 0.0468 0.0468 0.0468 ...
##  $ port_5000        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ windbearingsin   : num  -0.5878 -0.5878 -0.0698 -0.0698 -0.1045 ...
##  $ cloudcover       : num  1 1 1 1 0.97 ...
##  $ hldres_100       : num  13612 13612 13612 13612 13612 ...
##  $ natural_5000     : num  5172542 5172542 5172542 5172542 5172542 ...
##  $ hlres_300        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ hldres_300       : num  114994 114994 114994 114994 114994 ...
##  $ route_300        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ station_id       : num  16 16 16 16 16 16 16 16 16 16 ...
##  $ pressure         : num  1029 1029 1030 1030 1030 ...
##  $ route_500        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ hlres_500        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ hlres_100        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ pollutant        : Factor w/ 3 levels "NO2","PM10","PM2_5": 1 2 1 2 1 2 1 2 1 2 ...
##  $ industry_1000    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ zone_id          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ windbearingcos   : num  0.809 0.809 0.998 0.998 0.995 ...
##  $ windspeed        : num  6.55 6.55 4.47 4.47 4.11 ...
##  $ hldres_500       : num  357436 357436 357436 357436 357436 ...
##  $ hldres_1000      : num  1542650 1542650 1542650 1542650 1542650 ...

## [1] "Structure des données pour l'ensemble de test"

## 'data.frame':    300891 obs. of  33 variables:
##  $ ID               : int  448169 448170 448171 448172 448173 448174 448175 448176 448177 448178 ...
##  $ hlres_50         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ green_5000       : num  1678245 1678245 1678245 1678245 1678245 ...
##  $ hldres_50        : num  7725 7725 7725 7725 7725 ...
##  $ daytime          : num  72 72 73 73 74 74 75 75 76 76 ...
##  $ route_100        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ precipintensity  : num  0.6096 0.6096 0.0965 0.0965 0 ...
##  $ precipprobability: num  0.61 0.61 0.14 0.14 0 ...
##  $ hlres_1000       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ temperature      : num  9.49 9.49 8.22 8.22 7.58 ...
##  $ is_calmday       : Factor w/ 2 levels "False","True": 1 1 1 1 1 1 1 1 1 1 ...
##  $ route_1000       : num  15990 15990 15990 15990 15990 ...
##  $ roadinvdist      : num  0.0309 0.0309 0.0309 0.0309 0.0309 ...
##  $ port_5000        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ windbearingsin   : num  -0.5878 -0.5878 -0.0698 -0.0698 -0.1045 ...
##  $ cloudcover       : num  1 1 1 1 0.97 ...
##  $ hldres_100       : num  30902 30902 30902 30902 30902 ...
##  $ natural_5000     : num  1269882 1269882 1269882 1269882 1269882 ...
##  $ hlres_300        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ hldres_300       : num  278115 278115 278115 278115 278115 ...
##  $ route_300        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ station_id       : num  21 21 21 21 21 21 21 21 21 21 ...
##  $ pressure         : num  1029 1029 1030 1030 1030 ...
##  $ route_500        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ hlres_500        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ hlres_100        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ pollutant        : Factor w/ 3 levels "NO2","PM10","PM2_5": 1 2 1 2 1 2 1 2 1 2 ...
##  $ industry_1000    : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ zone_id          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ windbearingcos   : num  0.809 0.809 0.998 0.998 0.995 ...
##  $ windspeed        : num  6.55 6.55 4.47 4.47 4.11 ...
##  $ hldres_500       : num  772542 772542 772542 772542 772542 ...
##  $ hldres_1000      : num  2866112 2866112 2866112 2866112 2866112 ...
```

```
## [1] "Structure des données pour l'ensemble solution"
```

```
## 'data.frame':    448169 obs. of  2 variables:
##  $ ID    : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ TARGET: num  43 21 9 22 6 17 7 15 14 16 ...
```

Il y a 448169 observations dans l'ensemble d'apprentissage et 300891 observations dans l'ensemble de test, ce qui correspond à une proportion de 60% des données en apprentissage contre 40% des données en test. Ceci semble raisonnable. On remarque que la majorité des covariables sont de type numériques, ce qui nous permettra d'appliquer les techniques vues en cours. Il y a dans les ensembles de test et d'apprentissage deux covariables de type factorielles:

- is_calmday: variable booléenne (TRUE-FALSE) qui indique s'il s'agit d'un jour de week-end ou de vacances.
- pollutant: variable catégorielle présentant trois instances (NO2, PM10, PM2_5) qui indique le polluant considéré.

Décrivons désormais les variables de type numériques. On peut les rassembler en trois catégories. Tout d'abord les variables de type ID ou identité:

- ID: numéro de lignes du dataset (de 0 à 448168)
- daytime: valeur arbitraire qui décrit l'ordre temporel des données (un incrément de 1 correspond à 1h)
- zone_id: valeurs entre 0 et 5 qui correspondent à une ville donnée
- station_id: valeurs qui appartiennent à [16., 17., 20., 1., 18., 22., 26., 28., 6., 9., 25., 4., 10., 23., 5., 8., 11.]. Une station est liée à une zone_id. Plusieurs stations peuvent être liées à une même zone_id. On peut le comprendre comme un lieu de mesure dans une certaine ville (zone_id).
- pollutant : nom du polluant

Puis les variables numériques de type météorologiques et qui varient avec le temps:

- temperature : la température (double)
- windspeed : vitesse du vent (double)
- windbearing_cos : direction du vent (cos) (double)
- windbearing_sin : direction du vent (sin) (double)
- cloudcover : l'ensoleillement (double)
- precipitations_intensity : l'intensité des précipitations (double)
- precipitations_probability : les probabilités de précipitations (double)
- pressure : la pression (double)

Et enfin, il existe un certain nombre de variables statiques (dans le temps), que l'on qualifiera de territoriales et qui font référence aux lieux de mesure. Un buffer est un périmètre dessinée autour d'une position. Par exemple, le paramètres HLRES_50 correspond à la surface cumulée de terre résidentielle dans un cercel de 50 m de diamètre autour du point considéré.

- HLRES: terres résidentielles à faible densité (m2) – buffers de 50,100,300,500,1000
- HLDRES: HDRES + terres résidentielles à haute densité (m2) – buffers de 50,100,300,500,1000

- INDUSTRY: terres industrielles (m2) – buffer de 1000
- PORT: zones portuaires (m2) - buffer de 5000
- NATURAL: Terres semi-naturelles et forestières (m2) – buffer de 5000
- GREEN: parcs et espaces verts urbains + NATURAL (m2) – buffer de 5000
- ROUTE: distance cumulée de route dans le périmètre (m) – buffers de 100, 300, 500, 1000
- ROADINVDIST: distance inverse entre la station et la route la plus proche (1/m)

La covariable ID associe un unique entier à chaque combinaison unique de type (zone_id, station_id, pollutant, daytime). Ceci se comprend bien, puisque pour chaque ville, station associée, polluant considéré et ce à une date précise, le dataset renseigne des informations météorologiques et statiques (associée à un type de territoire autour d'un certain lieu). Nous allons pouvoir vérifier toutes ces suppositions en menant une EDA (Exploratory Data Analysis) plus approfondie. On remarque d'ors et déjà qu'une grande partie des données "statiques" présentent des observations de type "NotAvailable", qu'il faudra évidemment gérer par la suite. L'énoncé descriptif des données explique que les NA dans le dataset proviennent de deux raisons: * "No land use is encompassed within the buffer for land use data" (en anglais) * les données ne sont pas disponibles pour les paramètres météorologiques

Pour simplifier notre travail, on décide d'utiliser la fonction "attach" pour s'affranchir du nom du dataset considéré lorsqu'on souhaite accéder aux covariables du dataset d'entraînement. Évidemment, on ne le fait que pour ce dataset, afin de ne pas faire de confusion entre les covariables des datasets d'apprentissage et de test, qui présentent les mêmes noms pour leurs covariables.

3 EDA plus approfondie

3.1 Données manquantes

```
## [1] "Proportion de données manquantes pour le training set"
```

```
##                ID          hlres_50        green_5000         hldres_50
##           0.00000          84.56297          15.42610          49.19149
##           daytime         route_100   precipintensity precipprobability
##           0.00000          84.34541           0.00000           0.00000
##        hlres_1000       temperature        is_calmday        route_1000
##          18.86342           0.00000           0.00000           0.00000
##       roadinvdist         port_5000    windbearingsin        cloudcover
##           0.00000          49.95504           0.00000           0.00000
##        hldres_100      natural_5000         hlres_300        hldres_300
##          31.08069          31.93416          66.45216          31.08069
##         route_300        station_id          pressure         route_500
##          49.72655           0.00000           0.00000          18.86342
##         hlres_500         hlres_100         pollutant     industry_1000
##          66.45216          66.45216           0.00000          66.23461
##           zone_id    windbearingcos         windspeed        hldres_500
##           0.00000           0.00000           0.00000          31.08069
##       hldres_1000
##           0.00000
```

```
## [1] "Proportion de données manquantes pour le test set"
```

```
##                ID          hlres_50        green_5000         hldres_50
##           0.00000          83.57412          17.67218          48.13670
##           daytime         route_100   precipintensity precipprobability
```

```
##          0.00000         81.25268          0.00000          0.00000
##       hlres_1000      temperature       is_calmday       route_1000
##         18.77092          0.00000          0.00000          0.00000
##      roadinvdist        port_5000   windbearingsin       cloudcover
##          0.00000         53.94412          0.00000          0.00000
##       hldres_100     natural_5000        hlres_300        hldres_300
##         36.41950         34.33868         71.85692         36.41950
##        route_300       station_id         pressure        route_500
##         52.86898          0.00000          0.00000         18.77092
##        hlres_500        hlres_100        pollutant    industry_1000
##         71.85692         71.85692          0.00000         69.53548
##          zone_id   windbearingcos        windspeed        hldres_500
##          0.00000          0.00000          0.00000         36.41950
##      hldres_1000
##          0.00000
```

On remarque tout d'abord que les deux ensembles de test et d'entrainement présentent des proportions relativement identiques de données manquantes pour chacune de leurs différentes covariables. Suite à cette étude, on décide de supprimer les covariables "hlres_50" et "route_100" qui comptabilisent toutes deux plus de 84% de données manquantes, ce qui paraît difficilement gérable. Et ce pour les deux ensembles d'apprentissage et de test.

On va désormais s'occuper en priorité des variables telles que "hlres_500", "hlres_100" et "industry_1000" qui présentent un pourcentage de valeurs manquantes de l'ordre de 66% et des variables "hldres_50", "port_5000", "route_300" qui comptabilisent environ 50% de données manquantes. Enfin, il faut également gérer les variables "natural_5000", "hldres_300" et "hldres_500" (31% de données manquantes) et "green_5000", "hlres_1000" et "route_500" (entre 15% et 20% de données manquantes). Pour ce faire, nous allons nous fier aux explications annexes fournies par Plume Labs, qui laisse sous-entendre que les données manquantes pour ces variables descriptives des territoires en certains points font précisément référence à l'absence de tels territoires en ces points. Nous allons donc remplacer les "NA" données par des 0. On note par ailleurs qu'il n'y a pas de données manquantes pour les variables de type météorologiques. Ainsi, toutes les données manquantes de type "NA" sont à remplacer par des 0. Ceci est vrai pour les deux ensembles d'apprentissage et de test.

Les données sont désormais en partie nettoyées. Pour tenter de voir si l'on peut considérer les variables indiquées comme numériques/continues ou discrètes, analysons leurs différentes modalités. ## Nature des covariables

```
sapply(plume_training.data, function(x) length(unique(x)))
```

```
##               ID        green_5000         hldres_50           daytime
##           448169                15                 5             14256
##   precipintensity precipprobability       hlres_1000       temperature
##             1410               354                15              6071
##       is_calmday       route_1000      roadinvdist        port_5000
##                2               17               17                9
##    windbearingsin       cloudcover       hldres_100     natural_5000
##             3592              734                 5               12
##        hlres_300       hldres_300        route_300       station_id
##                6                5                 9               17
##         pressure        route_500        hlres_500        hlres_100
##             5774               15                 6                6
##        pollutant    industry_1000          zone_id    windbearingcos
##                3                6                 6             3580
##        windspeed       hldres_500      hldres_1000
##             3091               10                17
```

```
sapply(plume_test.data, function(x) length(unique(x)))
```

```
##                ID        green_5000         hldres_50           daytime
##            300891                10                 4             14243
##   precipintensity precipprobability        hlres_1000       temperature
##              1401               354                11              6067
##        is_calmday        route_1000       roadinvdist         port_5000
##                 2                12                12                 7
##    windbearingsin        cloudcover        hldres_100      natural_5000
##              3590               732                 5                 8
##         hlres_300        hldres_300         route_300        station_id
##                 4                 6                 6                12
##          pressure         route_500         hlres_500         hlres_100
##              5772                11                 4                 4
##         pollutant     industry_1000           zone_id    windbearingcos
##                 3                 4                 6              3578
##         windspeed        hldres_500       hldres_1000
##              3091                 6                12
```

Il semblerait que les variables de type "occupation des sols" (land use data) qui fournissent généralement une superficie en m^2 aient finalement peu de modalités pour qu'on puisse les considérer comme réellement continues. Cependant, cela semble assez logique vu le nombre de localisations considérées (6 modalités pour la variable "zone_id"), qui est assez faible. Ces variables statiques sont donc très souvent identiques pour différentes "IDs" du dataset. Une autre façon de se convaincre qu'il faut les considérer comme continue passe par l'observation des modalités des variables dans l'ensemble de test.

3.2 Comparaison test et train set

```
#Variable 'green_5000'
unique(plume_test.data$green_5000)
```

```
##  [1]  1678245  4013617  4087080  3716187 14427974  7231843  3596403
##  [8]  5823815  8499024        0
```

```
unique(green_5000)
```

```
##  [1]  5172542  2024936  3664739  4087080  2794815  5904352 10091554
##  [8] 10335997  2041264 11014751  2518087  9153980  8294676 19923668
## [15]        0
```

```
#Variable 'port_5000'
unique(plume_test.data$port_5000)
```

```
## [1]        0  4069418  2130139  4072668  3732988  2813698 10633674
```

```
unique(port_5000)
```

```
## [1]       0 4069418 4439135 5856246 2365708 2844116 1488916 4865436 6337544
```

```
#Variable 'hldres_50'
unique(plume_test.data$hldres_50)
```

```
## [1] 7725.425    0.000 5963.704 7667.613
```

```
unique(hldres_50)
```

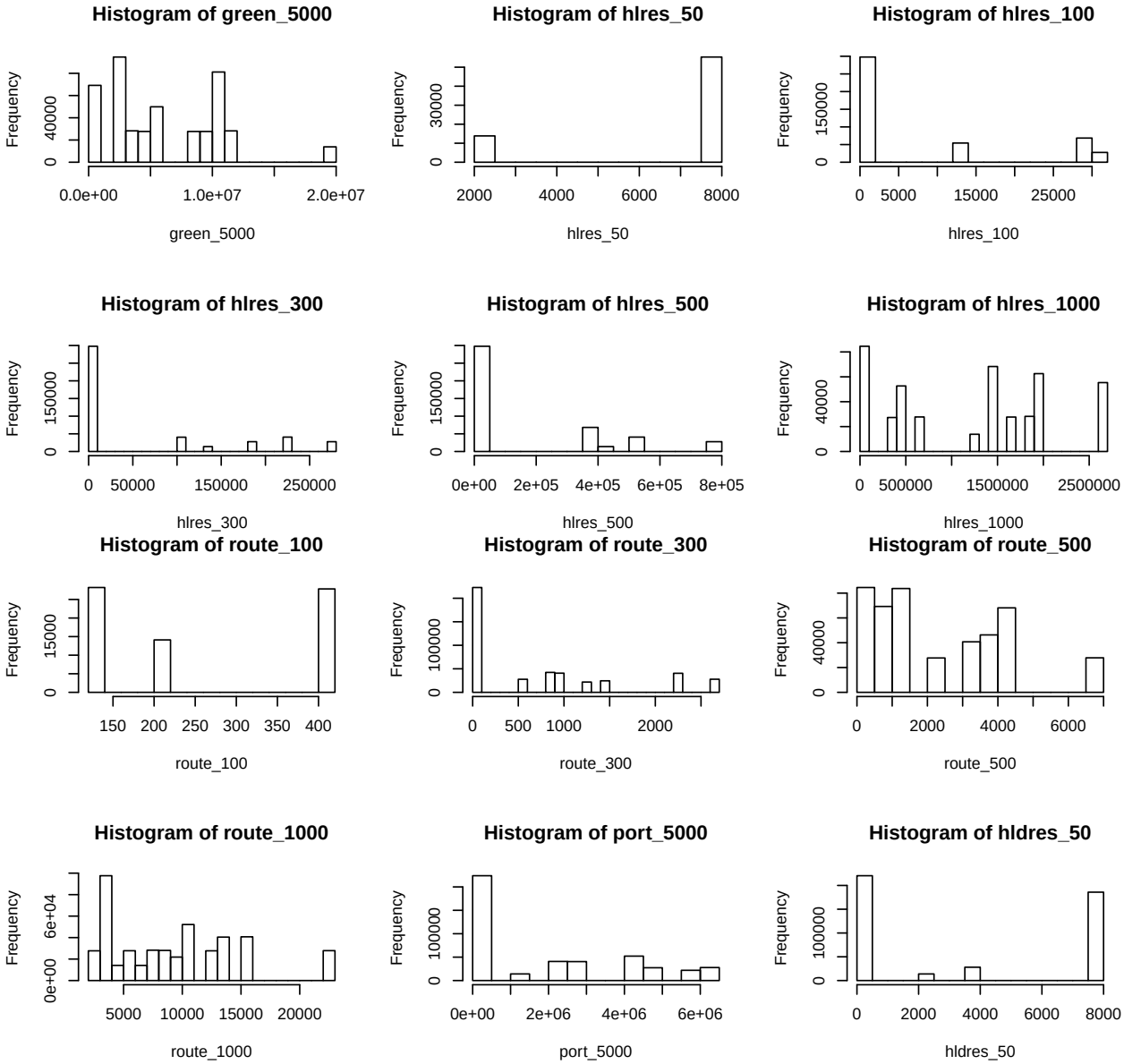
```
## [1] 3755.190 7725.425    0.000 7725.425 2217.580
```

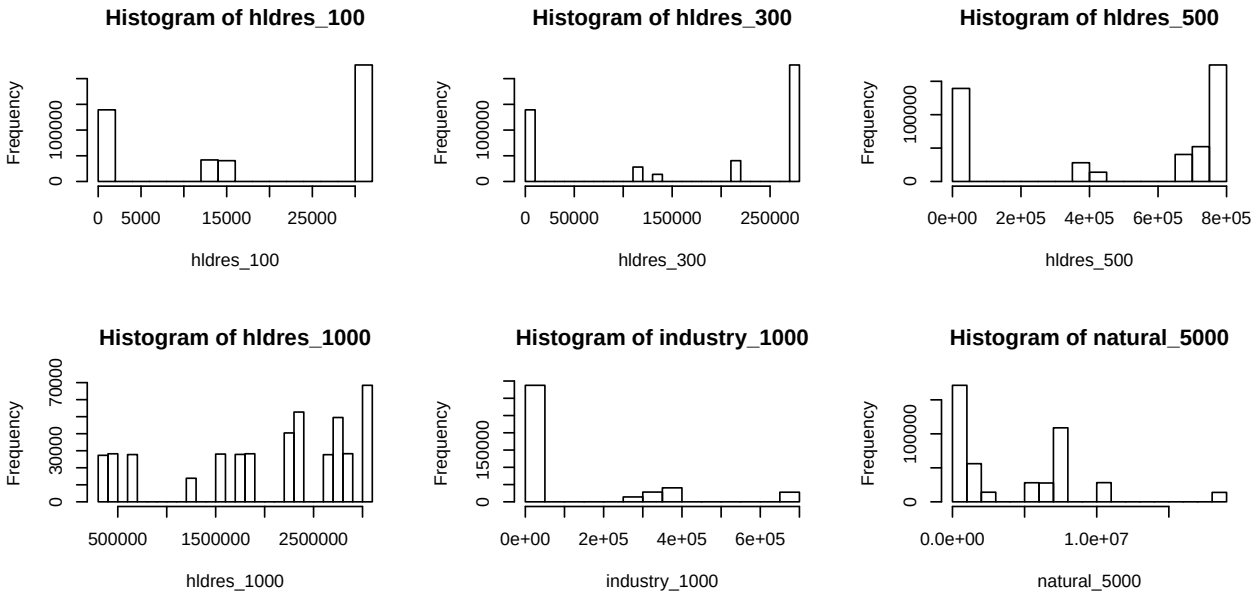
On remarque effectivement que les modalités ne se recoupent généralement pas entre le test et le train set, ce qui nous conforte à considérer ces variables comme continues pour notre problème, afin d'être sur de bien généraliser notre prédicteur/régresseur à de nouveaux datasets.

Désormais, continuons d'analyser les covariables et leurs interactions. Cela nous permet aussi de regarder si nos deux ensembles d'apprentissage et de test sont à peu près construits de la même façon.

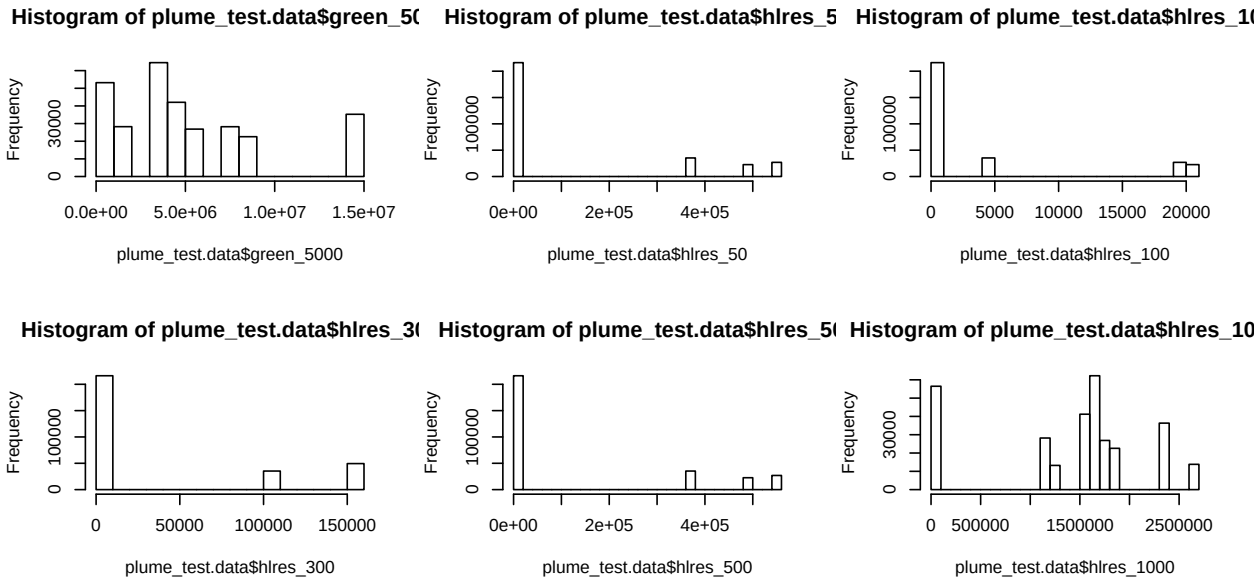
3.3 Analyse univariée des covariables

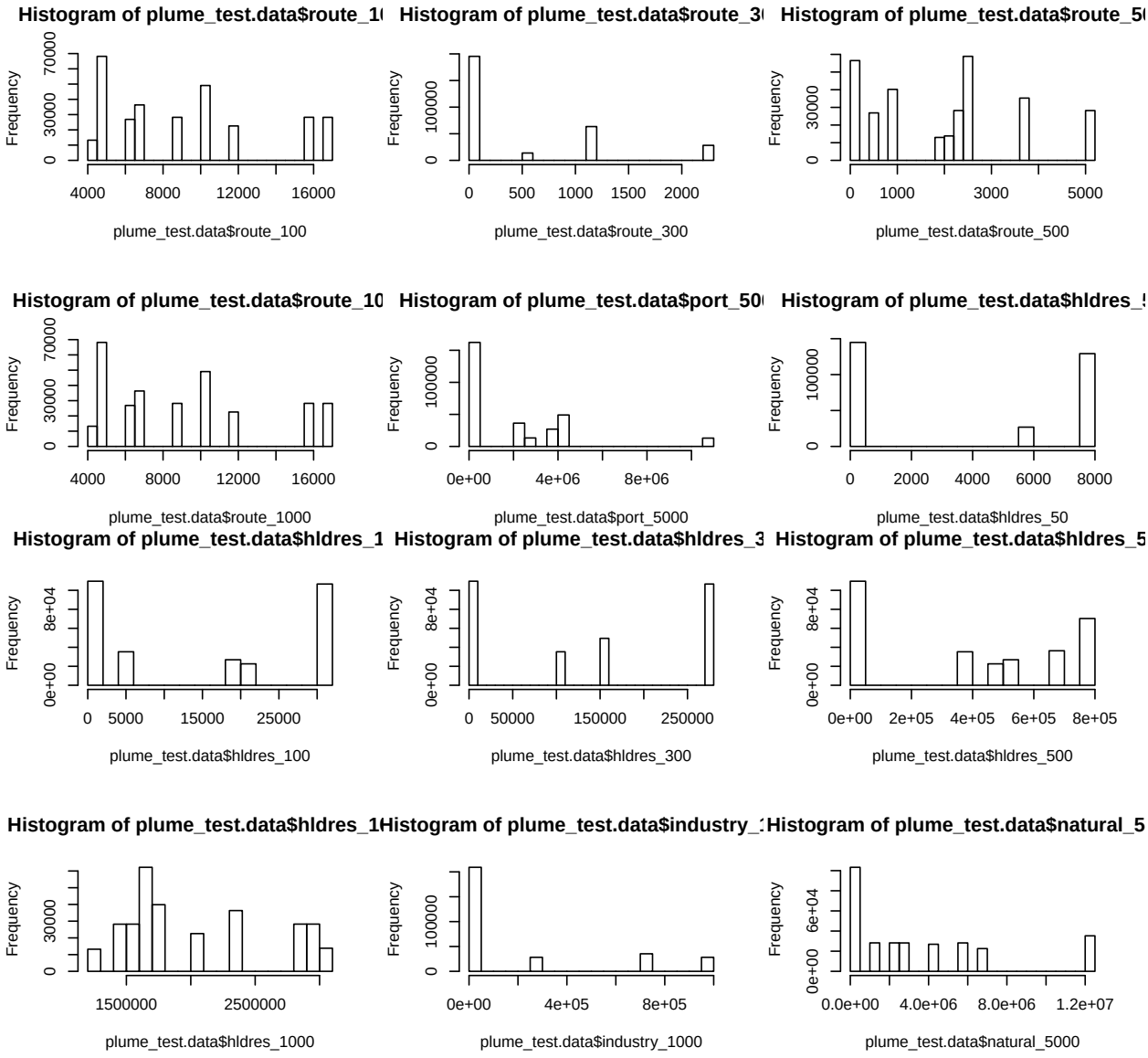
3.3.1 Variables statiques territoriales pour le train set



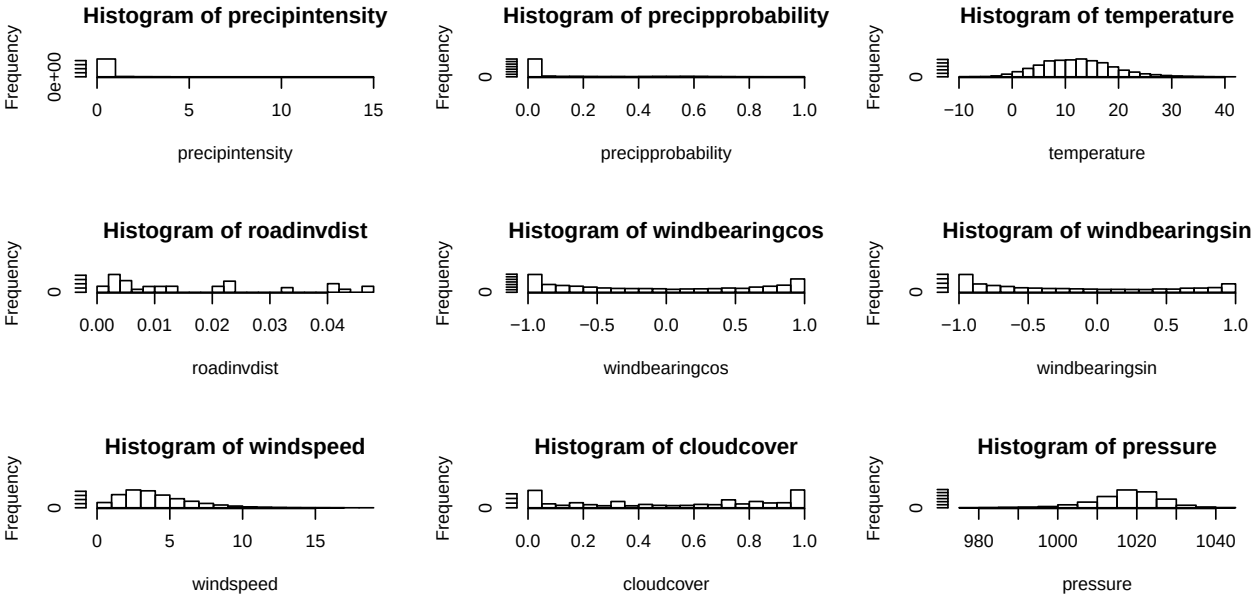


3.3.2 Variables statiques territoriales pour le test set

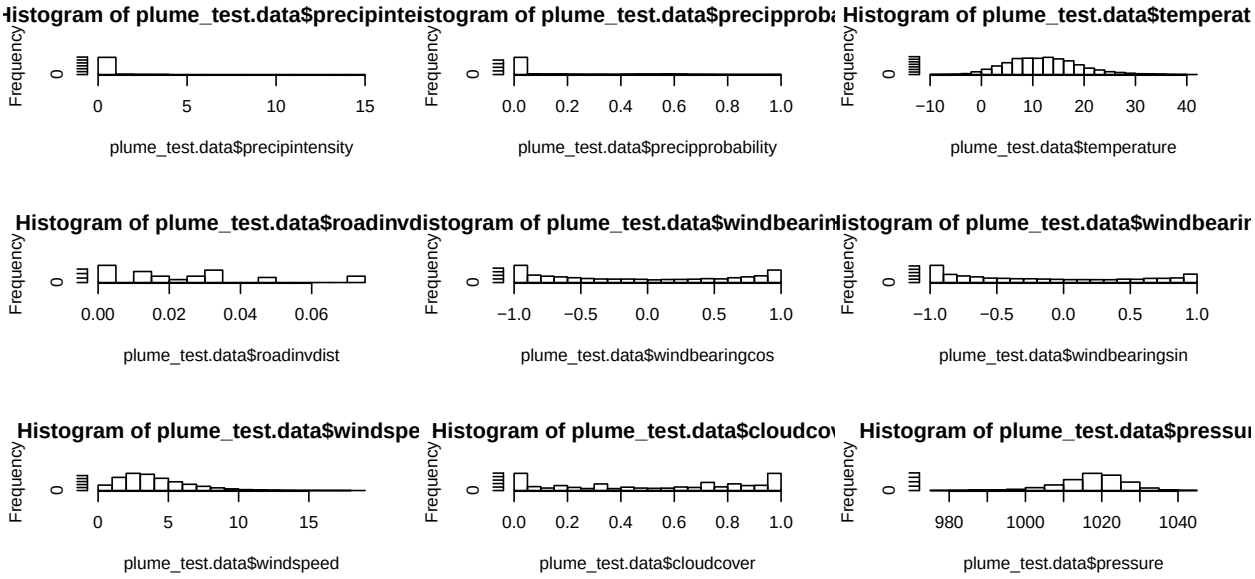



Histogram of plume_test.data$route_1(
Frequency
0
30000
70000
4000
8000
12000
16000
plume_test.data$route_100
Histogram of plume_test.data$route_3(
Frequency
0
100000
0
500
1000
1500
2000
plume_test.data$route_300
Histogram of plume_test.data$route_5(
Frequency
0
30000
0
1000
3000
5000
plume_test.data$route_500
Histogram of plume_test.data$route_10
Frequency
0
30000
70000
4000
8000
12000
16000
plume_test.data$route_1000
Histogram of plume_test.data$port_50(
Frequency
0
100000
0e+00
4e+06
8e+06
plume_test.data$port_5000
Histogram of plume_test.data$hldres_!
Frequency
0
100000
0
2000
4000
6000
8000
plume_test.data$hldres_50
Histogram of plume_test.data$hldres_1
Frequency
0e+00
8e+04
0
5000
15000
25000
plume_test.data$hldres_100
Histogram of plume_test.data$hldres_3
Frequency
0e+00
8e+04
0
50000
150000
250000
plume_test.data$hldres_300
Histogram of plume_test.data$hldres_5
Frequency
0e+00
8e+04
0e+00
2e+05
4e+05
6e+05
8e+05
plume_test.data$hldres_500
Histogram of plume_test.data$hldres_1(
Frequency
0
30000
1500000
2500000
plume_test.data$hldres_1000
Histogram of plume_test.data$industry_:
Frequency
0
100000
0e+00
4e+05
8e+05
plume_test.data$industry_1000
Histogram of plume_test.data$natural_5
Frequency
0e+00
6e+04
0.0e+00
4.0e+06
8.0e+06
1.2e+07
plume_test.data$natural_5000

3.3.3 Variables météorologiques pour le train set

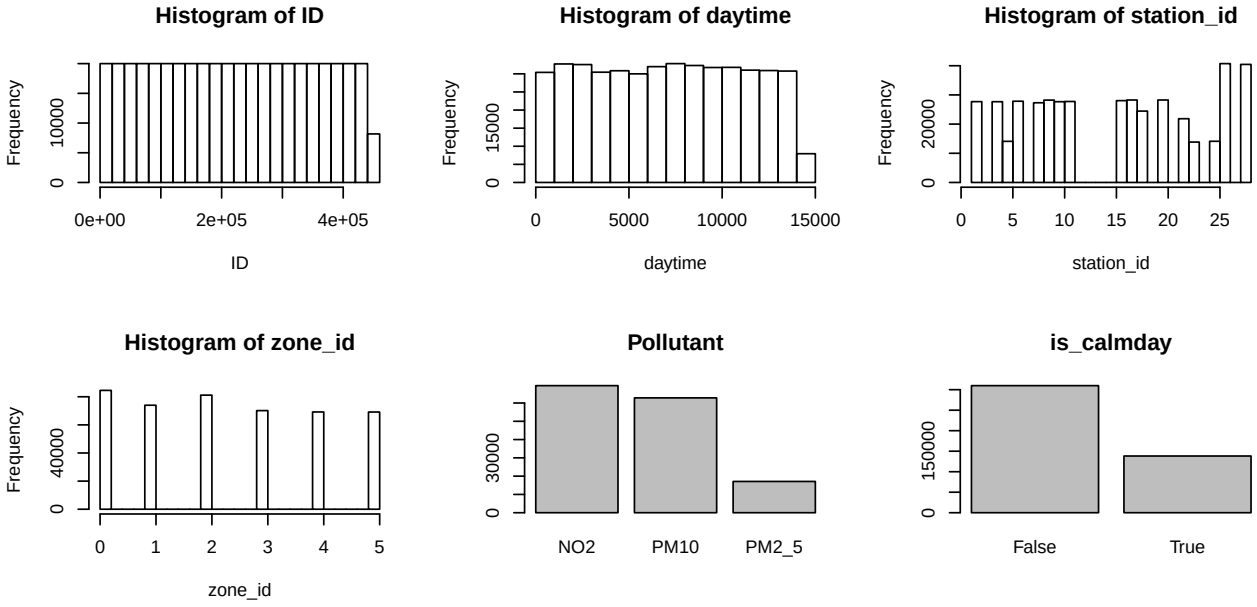


3.3.4 Variables météorologiques pour le test set

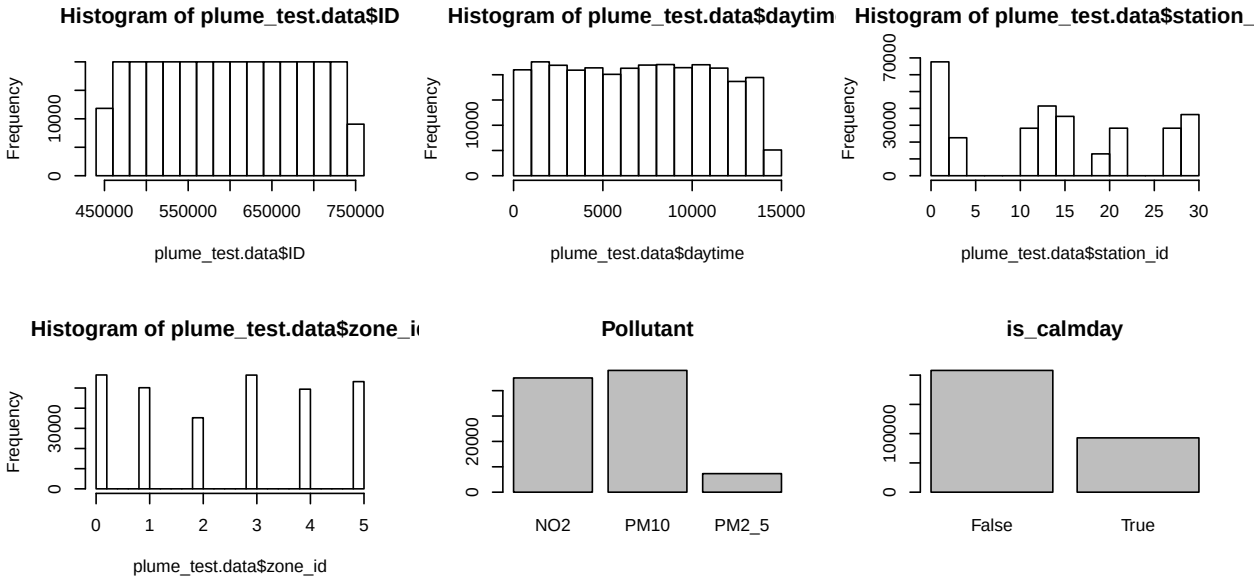


Les premiers graphiques qui montrent les supports des covariables des train et test set laissent sous-entendre qu'elles sont globalement réparties de façon identique entre train et test set. Il est désormais plus intéressant de s'attarder sur les covariables de type identité.

3.3.5 ID variables pour le train set



3.3.6 ID variables pour le test set



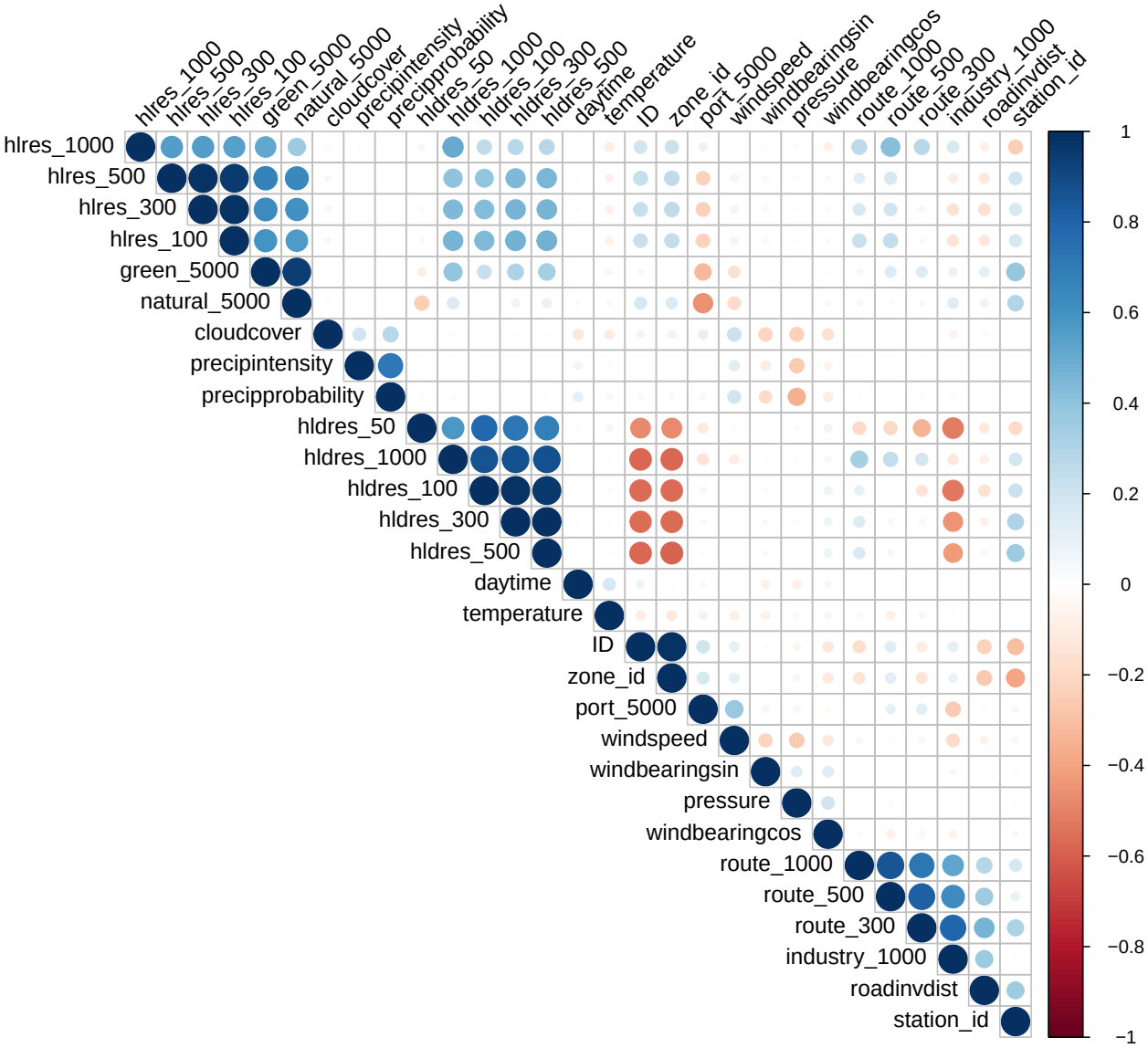
On remarque que les mêmes périodes de temps globalement (covariables daytime) sont partagées par le train et le test set. Il ne s'agit donc pas d'estimer la concentration d'un polluant pour un temps ultérieur, mais plus de bien prendre en compte les interactions temporelles déjà existantes pour estimer la concentration d'un polluant en un temps pour lequel on possède des informations précises. Par ailleurs les covariables "zone_id", "is_calmday" et "pollutant" présentent globalement les mêmes répartitions. Il est par contre primordial de remarquer que la station_id ne semble pas partagée de la même façon dans l'ensemble d'apprentissage et de test. C'est une remarque importante, puisque cette observation peut être la source de futurs problèmes et peut aboutir sur des moyens d'améliorer considérablement notre prédiction.

3.4 Analyse multivariée des covariables

On s'intéresse désormais à l'analyse multivariée des différentes covariables. Pour cela, une visualisation intéressante est la matrice de corrélation entre les covariables, présentées ci-après.

```
## Loading required package: corrplot
```

```
## corrplot 0.84 loaded
```



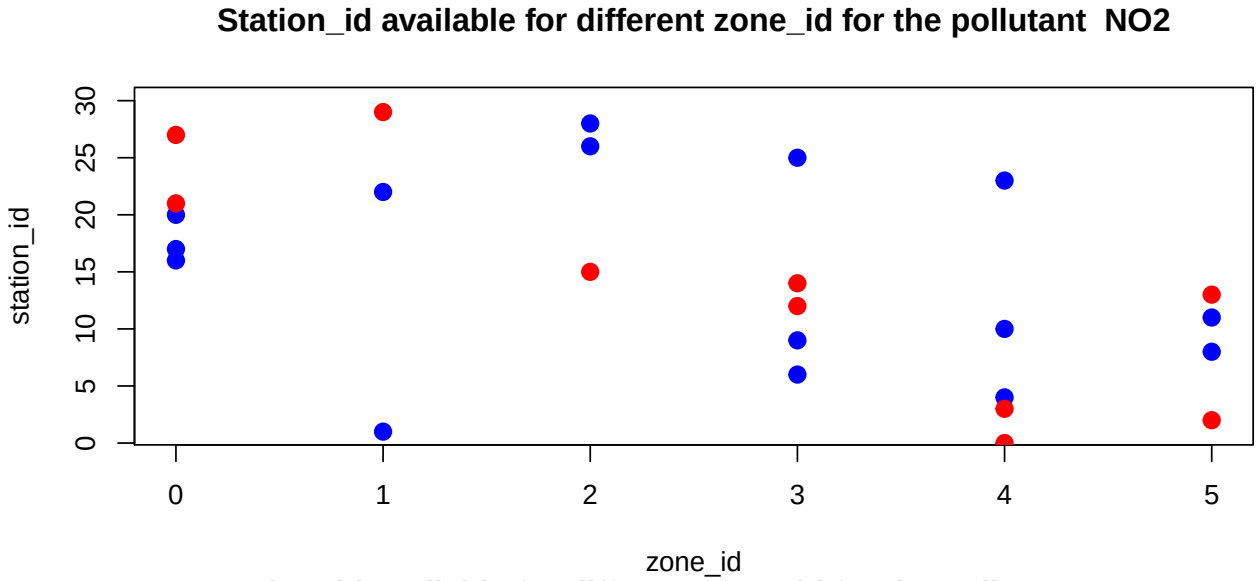
L'observationd de cette matrice de corrélation entre les covariables nous permet de faire plusieurs remarques intéressantes:

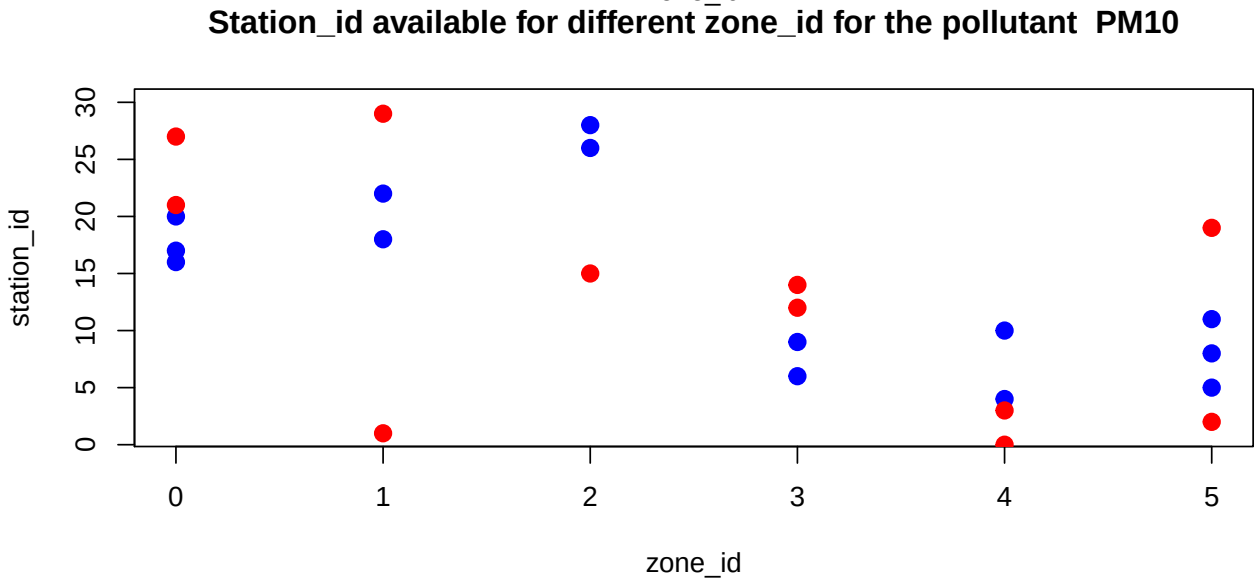
- Les variables de types territoriales sont très corrélées entre elles, mais bien moins avec les variables de type météorologiques. Il est en de même pour les variables de type météorologiques. On comprend bien qu'elles n'ont pas tout à fait le même sens, et que les premières sont statiques dans le temps et communes à une même station_id, tandis que ce n'est pas le cas des secondes qui sont fortement dépendantes du temps. A ce stade, on peut se dire que dans une analyse future, il pourrait être intéressant de séparer notre jeu de données en deux data set : l'un contenant les features météorologiques et l'autre contenant les features territoriaux et de mettre en place des stratégies d'apprentissage distinctes.

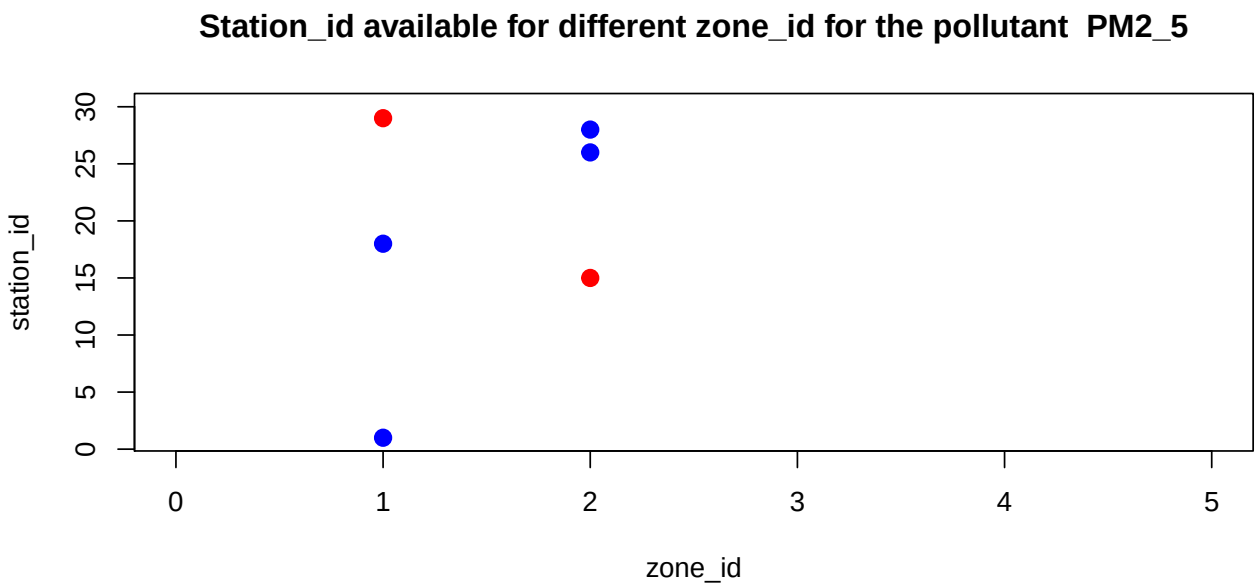
- Par ailleurs, si l'on s'intéresse au feature "roadinvdist" (inverse de la distance à la plus proche route (1/m)), il semble très corrélé aux features "route_300", "route_500", "route_1000" (distance cumulée de route dans le périmètre (m)) et on comprend que l'on pourrait créer un nouveau feature adimensionné tel que road = roadinvdist * route_buffer.

3.5 Investigation plus poussée des différences entre train set et test set

On a effectivement remarquer que les variables "station_ID" ne présentent pas les mêmes réalisations entre le train set et le test set. Ceci peut-être à l'origine de problème futurs, ou le signe qu'il faut repenser notre façon de travailler. On examine donc la répartition des valeurs pour station_id en fonction des zone_id, pour les données du train et du test set séparément.







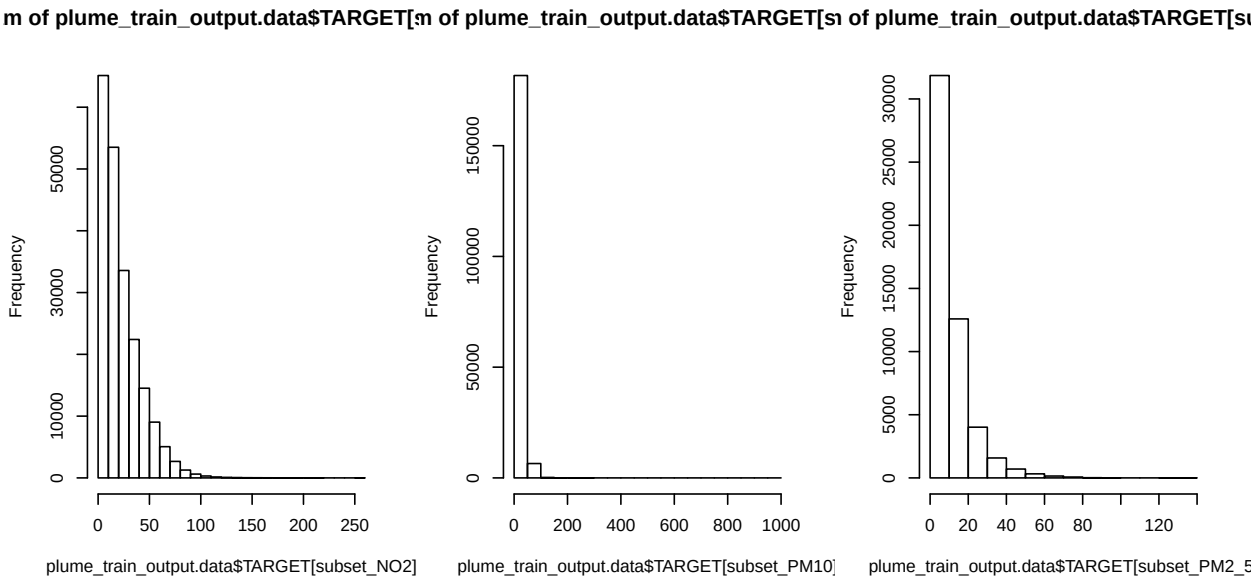
Sur les graphiques précédents, la couleur bleue fait référénces aux données d'apprentissage, tandis que le rouge fait référence aux données de test. On remarque grâce aux graphiques précédents qui présentent les station_id disponibles pour chaque zone_id (ville) pour chaque polluants que les données ne se recoupent jamais entre train et test set. Cela explique ce que l'on observe précèdemment. Le problème est finalement un peu différent : il s'agit, pour chaque polluant, et pour chaque ville de prévoir la concentration en des stations précises, sachant que l'on connait les concentrations en d'autres lieux/stations de cette même ville.

Cette observation est très importante et elle sera prise en considération par la suite afin de tenter d'approfondir notre modèle.

4 Test de la régression linéaire

Pour effectuer une régression linéaire il nous faut tout d'abord retravailler la forme des ensembles afin d'intégrer la quantité que l'on souhaite prédire dans le dataset, à savoir la concentration des différents polluants considérés.

4.1 Encodage des dummy variables



Une des premières étapes à mettre en place afin d'appliquer correctement un modèle de régression linéaire est la création de variables "dummy" afin d'encoder numériquement les variables catégorielles, qui sont ici "is_calmday" et "pollutant". Ci-dessus, nous avons tracé les histogrammes des trois réalisations possibles pour chacun des polluants, de la variable "TARGET". Au vu des supports de ces différentes réalisations, on décide d'incorporer une notion de rang/d'ordre dans notre façon d'encoder les niveaux de la variable "pollutant": NO2 à 0 car ses concentrations sont les plus faibles, PM2_5 à 1 car ses concentrations sont intermédiaires et PM10 à 2 car ses concentrations sont en moyenne plus élevées. On fait cela pour le train et test set, bien entendu.

On met en place une régression linéaire classique.

4.2 Régression linéaire classique par Moindres Carrés.

```
lm.fit <- lm(TARGET~.,data = plume_tot_training[,-1])
lm.summary<-summary(lm.fit)
```

On s'assure avant d'analyser les résultats que l'on peut bien utiliser ces techniques classiques de régression linéaire (matrice de rang plein et hypothèse des résidus normalisés, hétéroscédasticités, homogénéité ...).

```
## Loading required package: Matrix
```

```
## [1] "Le range de la matrice obtenu vaut 28"
```

Notre matrice des covariables n'est donc pas de rang plein et il faut donc utiliser d'autres modèles de régression linéaire plus robustes. Dans un premier temps, nous allons utiliser la régression Ridge par exemple.

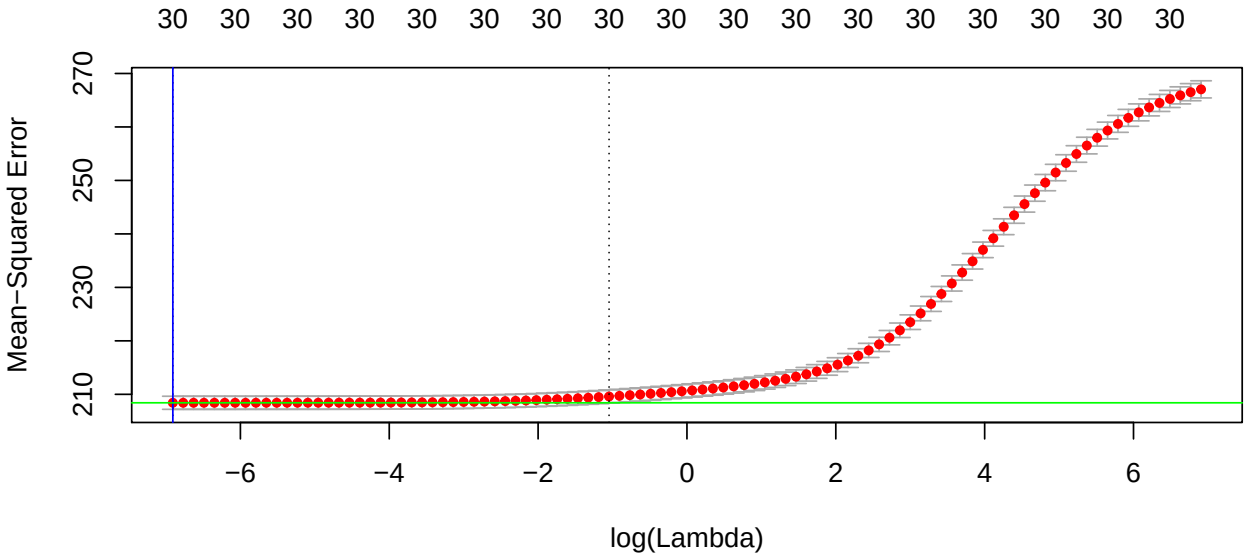
4.3 Régression Ridge

La régression Ridge peut s'effectuer avec le package glmnet (option alpha=0). Cependant, il faut d'abord créer un objet de type matrice contenant les prédicteurs. Pour pouvoir analyser la qualité de notre modèle, nous allons devoir le tester sur un ensemble de test. A cet égard, il est important de noter que l'ensemble de test fourni par les organisateurs du challenge ne peut être utilisé que pour soumettre une solution sur la plateforme du challenge, qui jugera elle-même de la performance de notre modèle. Il est donc impératif de

créer un ensemble de validation supplémentaire, issu de l'ensemble d'apprentissage fourni par la plateforme et pour lequel nous connaissons les valeurs TARGET. Avant d'utiliser la régression ridge, il nous faut donc partitionner notre jeu de donnée plume_tot_training en un ensemble d'apprentissage et de validation. Vu la quantité de données disponibles, nous allons réserver 30% des données pour le test et 70% pour l'apprentissage.

Pour la régression Ridge, l'estimation des coefficients dépend du choix du paramètre λ. On peut par exemple visualiser, pour chaque coefficient, une trajectoire de son estimateur en fonction de λ.

```
## [1] "mybestlam = 0.001 "
```



Il est important de noter que nous avons sélectionner une grille acceptable pour λ telle que $\lambda \in [10^{-3}; 10^{3}]$. J'ai également tenté de diminuer la borne min de cet intervalle, et la validation croisée sélectionnait systématiquement la valeur la plus faible possible pour λ. Cette observation n'est pas satisfaisante, puisque le modèle choisi a tendance à être complexe (faible λ) et on augmente donc le risque de sur-apprentissage.

```
## [1] "L'erreur de prédiction sur l'échantillon d'apprentissage vaut"
```

```
## [1] "(root mean square normalized) 208.385583423988"
```

Nous avons spécifiquement tenté différentes type de régression plus robustes : Ridge, LASSO (qui fait de la sélection de modèles) et Elastic-Net (alpha = 0.5).

```
## [1] "L'erreur de prédiction sur l'échantillon de test vaut "
```

```
## [1] "(root mean square normalized) 215.51022251217"
```

Ces erreurs semblent relativement acceptables et nous pouvons donc désormais tenter de soumettre notre solution sur le site du challenge. Pour ce faire, il est important d'utiliser la totalité des données étiquetées à notre disposition pour entraîner le prédicteur.

Les résultats obtenus avec nos ensembles d'apprentissage et de test (et non l'ensemble de test du challenge) sont présentés ci-dessous:

Méthode	Mean square error train	Mean square error test
Régression Ridge	211.8	207.5
Régression LASSO	212.7	219.7
Régression Elastic-Net	211.8	207.5

Nous présentons désormais les résultats obtenus lors de la soumission de nos résultats sur le site du challenge, en comparaison avec le benchmark initial de Plume Labs et le meilleur score du leaderboard:

Méthode	Public score	Private score
Régression Ridge	373.6	353.3
Plume Labs (benchmark)	501.3	480.1
Meilleur score du leaderboard	193.2	178.8

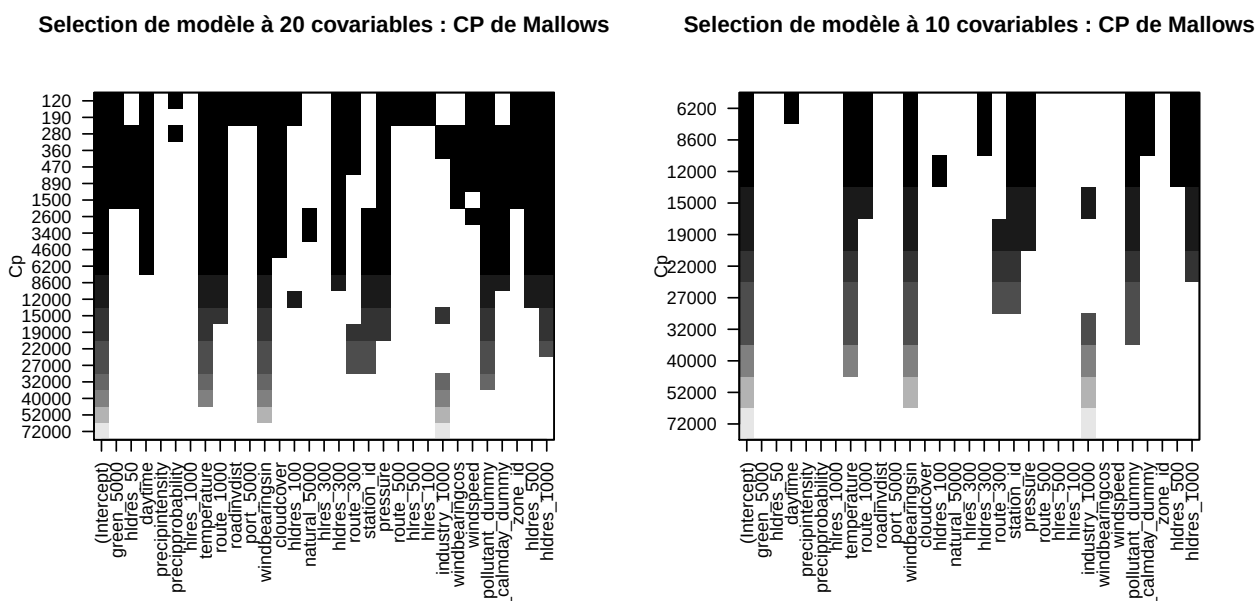
5 Critères pénalisés de sélection de modèles

Comme on a pu le voir, nos données présentent un assez grand nombre de covariables et il peut être intéressant de mettre en place des critères pénalisés pour sélectionner certaines covariables. Même si une sélection de modèle n'aboutit pas nécessairement, dans le sens où il est parfois plus intéressant de garder la totalité des covariables, il sera toujours intéressant d'analyser quelles covariables spécifiques sont sélectionnées. Nous présentons ainsi les résultats des critères de sélection du Cp de Mallows et du critère BIC avec recherche exhaustive et ce pour des modèles à 20 et 10 covariables retenues.

5.1 Critère du Cp de Mallows

```
## Reordering variables and trying again:
```

```
## Reordering variables and trying again:
```



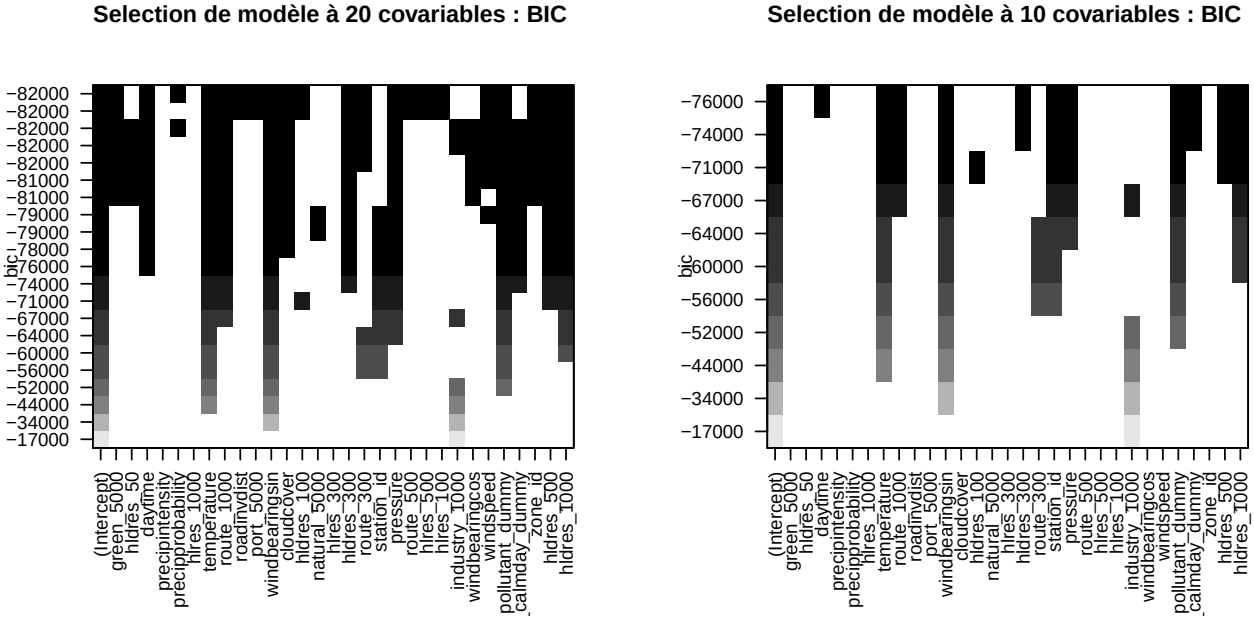
5.2 Critère BIC

On effectue désormais des tests avec le critère BIC.

On regarde désormais ce que l'on obtient en implémentant le critère BIC.

```
## Reordering variables and trying again:
```

```
## Reordering variables and trying again:
```



Pour ces quatre modèles, nous avons pu mettre en place une validation croisée (comme précèdemment) afin de déterminer le meilleur λ et déterminer les mean square error en conséquences. Nous avons également pu examiner quelles variables étaient rejetées en priorité. Les résultats sont présentés dans le tableau ci-dessous:

Méthode de sélection de variables	MSE train	Mse test	Dimension modèle max	Dimension modèle choisie	Variables rejetées
C_p de Mallows	213.7	206.3	20	20	territoriales
C_p de Mallows	217.5	213.2	10	10	territoriales (majorité) + météo
BIC	212.8	205.4	20	20	territoriales
BIC	216.0	205.4	10	10	territoriales (majorité) + météo

Il est désormais très important de remarquer que la différence de MSE atteinte avec notre test set (environ 200) et le test des challengeurs (plutôt 370) provient du test set artificel que nous avons créer à partir des données d'apprentissage. Effectivement, nous avons choisi 30% des données de façon aléatoire, ce qui ne garantit absolument pas que les train et test set ne partagent pas de station_id identiques. Il faudrait en pratique s'assurer que notre test set est construit comme celui surlequel on souhaite généraliser notre modèle. C'est pour cette raison que nos résultats sont meilleurs que ceux qui nous ont été communiqués par la plateforme du challenge.

6 Amélioration de nos résultats

Les résultats obtenus ne sont pas si mauvais, mais il est encore possible de les améliorer. A ce propos, nous allons émettre quelques pistes futures à investiguer.

- Regrouper les données par polluant et entraîner trois régresseurs différents. Cette idée à l'avantage de se débarasser du problème d'encodage des dummy variables et fait complètement sens, puisqu'à priori, les concentrations de ces trois polluants ne devraient pas être liées.

- Créer de nouveaux features/ de nouvelles covariables que l'on pourrait qualifier de "lag" variables afin de capturer les dépendances temporelles pour les données de type météorologiques. C'est une pratique tout à fait courante lorsqu'on travaille sur un jeu de donnée type série temporelle.
- Des suites de notre EDA, il semblerait que notre jeu de données puisse bien se scinder en deux catégories de variables : les variables météorologiques et les variables territoriales statiques. Les secondes sont plus complexes à comprendre et gérer puisqu'elles sont plutôt de type discrètes que continues numériques et qu'elles sont communes à toutes les station_id et constantes dans le temps, donc en un sens, elles apportent moins d'informations que les variables météorologiques qui varient temporellement. La séparation des variables en deux ensemble distincts est une piste intéressante à exploiter.
- Il faut également se rappeler de la répartition de la variables "station_id" qui n'est pas commune au train et test set. Une façon intéressante de se débarasser de cette difficulté est de moyenner certains résultats sur les différentes station_id disponibles. Nous en discuterons plus précisément par la suite.

6.1 Travail avec les features météorologiques uniquement

Dans un premier temps, on ne prend en compte que les features météorologiques, en séparant en trois cluster les prédictions pour chaque polluant. Cette idée provient du fait que la sélection de variables laissait sous-entendre que les features météorologiques étaient plus importants que les territoriaux. Par soucis de clarté, le code n'est pas affiché ici, mais il est bien entendu disponible sur le fichier Rmarkdown. Il est intéressant de noter que l'on obtient, avec cette approche un meilleur résultat pour la mean square error que lorsque la totalité du jeu de données était considérée.

Public score	Private score
354	332

6.2 Travail avec l'ensemble des données

Une nouvelles alternative afin de prendre en compte les features territoriaux se pose comme suit:

- Mise en place d'un clustering avec trois prédicteurs pour chaque polluants ($NO_2, PM_{10}, PM_{2.5}$)
- On prédit la concentration à l'aide des variables météorologiques uniquement (comme précèdemment), pour chaque station_id, daytime et zone_id (et chaque polluant) avec une régression ridge.
- Ensuite, en utilisant les features territoriaux, on tente de prédire (régression ridge) TARGET - TARGET_mean_station_id, où TARGET_mean_station_id est la moyenne des concentrations pour chaque polluant par station_id.
- On somme ensuite les deux contributions précedemment prédites pour donner notre prédiction finale de la concentration du polluant.

Encore une fois, par soucis de clarté, le code n'est pas présenté dans ce rapport, mais il est bien sur accessible depuis sur le fichier Rmarkdown. Les résultats obtenus sont les suivants:

Public score	Private score
358.7	336.4

Il est étrange de remarquer que le score ne s'est pas amélioré par rapport au cas précédent, où l'on ne considérait que les covariables météorologiques. Je n'ai pas d'explications précises à ce propos, puisque je pensais au contraire que la deuxième prédiction basée sur les features territoriaux (TARGET - TARGET_mean_station_id) nous permettrait de rectifier la première prédiction afin de mieux prendre en compte

le feature “station_id”.

6.3 Création de lag variables pour les features météorologiques

Dans un troisième temps, nous avons décidé d’ajouter des lags covariables pour certains features, explicitement: “temperature”, “pressure”, “cloudcover”, “windspeed” moyenné, pour chaque polluant spécifiques sur différentes périodes (pour toutes les zone_id et station_id) :

- le quart de journée (6 heures)
- la demi-journée (12 heures)
- la journée (24 heures)
- la semaine (168 heures)
- le mois (672 heures)
- le trimestre (3 mois, 1800 heures)
- le semestre (6 mois, 3600 heures)

Nous ne pensions pas initialement ajouter autant de lag, et nous avons commencé par le lag au jour (24h) et à la semaine (168h), mais en ajoutant les lags cités ci-dessus, nous nous sommes rendus que les prédictions s’amélioraient (légèrement). Les résultats obtenus sont les suivants sur le test set du challenge sont les suivants:

Public score	Private score
338.0	315.8

On remarque que les résultats obtenus sont relativement meilleurs. Peut-être pas autant qu’espéré vu le nombre de lags qui ont été rajoutés.

7 Conclusion

Ce projet s’est avéré très intéressant et enrichissant, tant du point de vue de la compréhension claire d’un jeu de données, indispensable au bon démarrage d’un projet de machine learning que du point de vue des différentes méthodes statistiques applicables. Pour conclure, je souhaiterais revenir sur deux points plus précisément. Tout d’abord, il semblerait que je n’ai pas réussi à bien prendre en compte les features territoriaux dans mon étude, puisque mes résultats sont meilleurs lorsque je n’en tiens pas compte. Il semble clair qu’ils jouent un rôle particulier avec le feature “station_id”, mais je ne vois pas, à ce jour de bonne et différente stratégie pour les relier de façon intelligente. On peut également ajouter qu’il aurait été intéressant de faire une analyse en composantes principales en complément, même si, ici, il semblerait que le modèle soit au contraire plus performant si on lui ajoute/crée de nouveaux features. C’est ce qu’on a pu observé avec les créations de lag features entre autres. Enfin, avec plus de temps, j’aurais souhaité me pencher sur l’utilisation de techniques de machine learning plus avancées telles que les SVR (support vector regression) ou les techniques d’ensembling (Random Forest et Gradient Boosting Decision Trees) qui permettent de capturer des non-linéarités, contrairement à ces modèles de régression linéaire plus basiques.