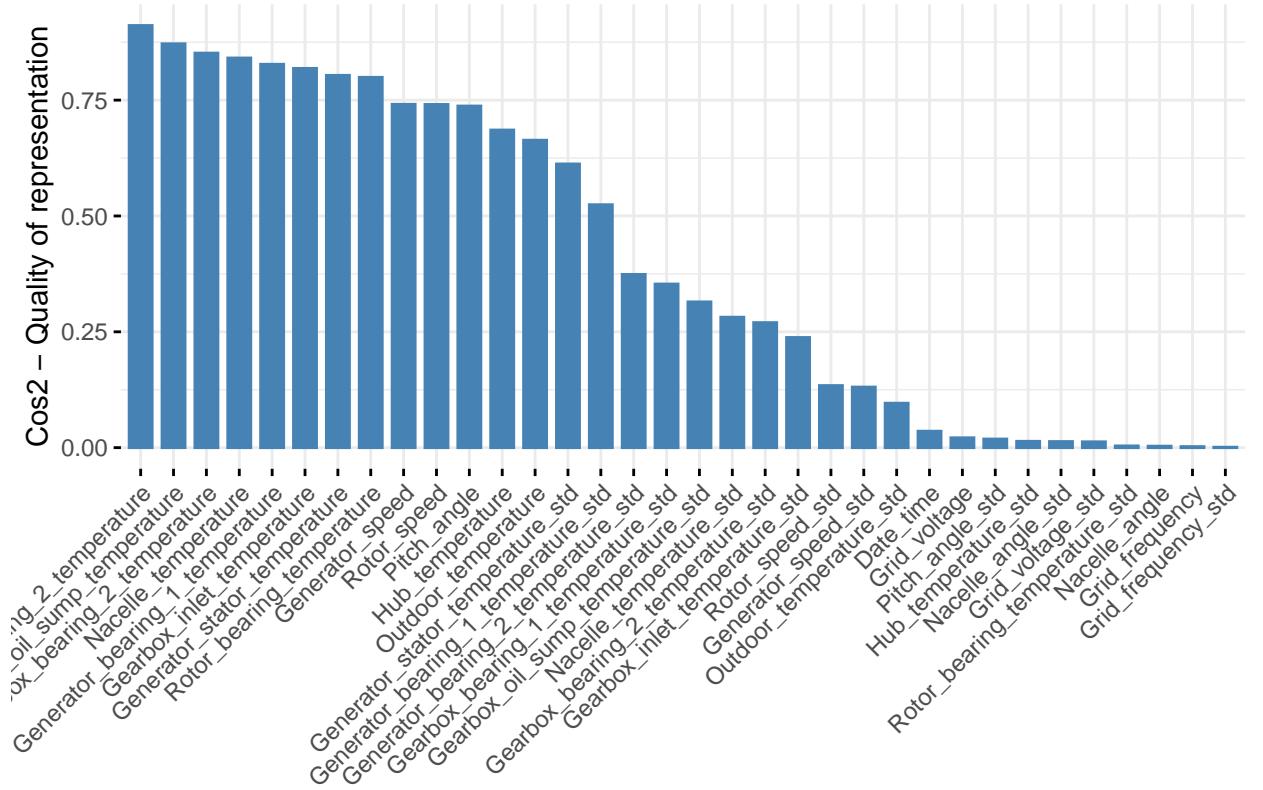


## Cos2 of variables to Dim-1-2



Les variables `Generator_bearing_1_temperature`, `Gearbox_oil_sump_temperature` et `Nacella_temperature` ont un  $\cos^2$  élevé sont bien représentées par les axes principaux 1-2. En revanche, les variables `Grid_voltage`, `Nacella_angle` et `Grid_frequency` ont un  $\cos^2$  très faible, elles sont donc mal représentées par les axes principaux 1-2.

- Contribution des variables

Dim.1\$quanti	Corrélation
Generator_speed_std	0.76032684
Rotor_speed_std	0.75877857
Gearbox_bearing_1_temperature_std	0.62360100
Generator_stator_temperature	-0.087450697
Gearbox_oil_sump_temperature	-0.106711545
Nacelle_angle_std	-0.113520239

Dim.2\$quanti	Corrélation
Gearbox_oil_sump_temperature_std	0.524247615
Rotor_speed	0.468734326
Generator_speed	0.468601377
Pitch_angle_std	-0.136354131
Pitch_angle	-0.203310594
Gearbox_inlet_temperature	-0.206463311

Dim.3\$quanti	Corrélation
Generator_speed_std	0.76032684
Rotor_speed_std	0.75877857
Gearbox_bearing_1_temperature_std	0.62360100
Gearbox_inlet_temperature	-0.10513183
Grid_voltage	-0.10792038
Grid_frequency	-0.10919804

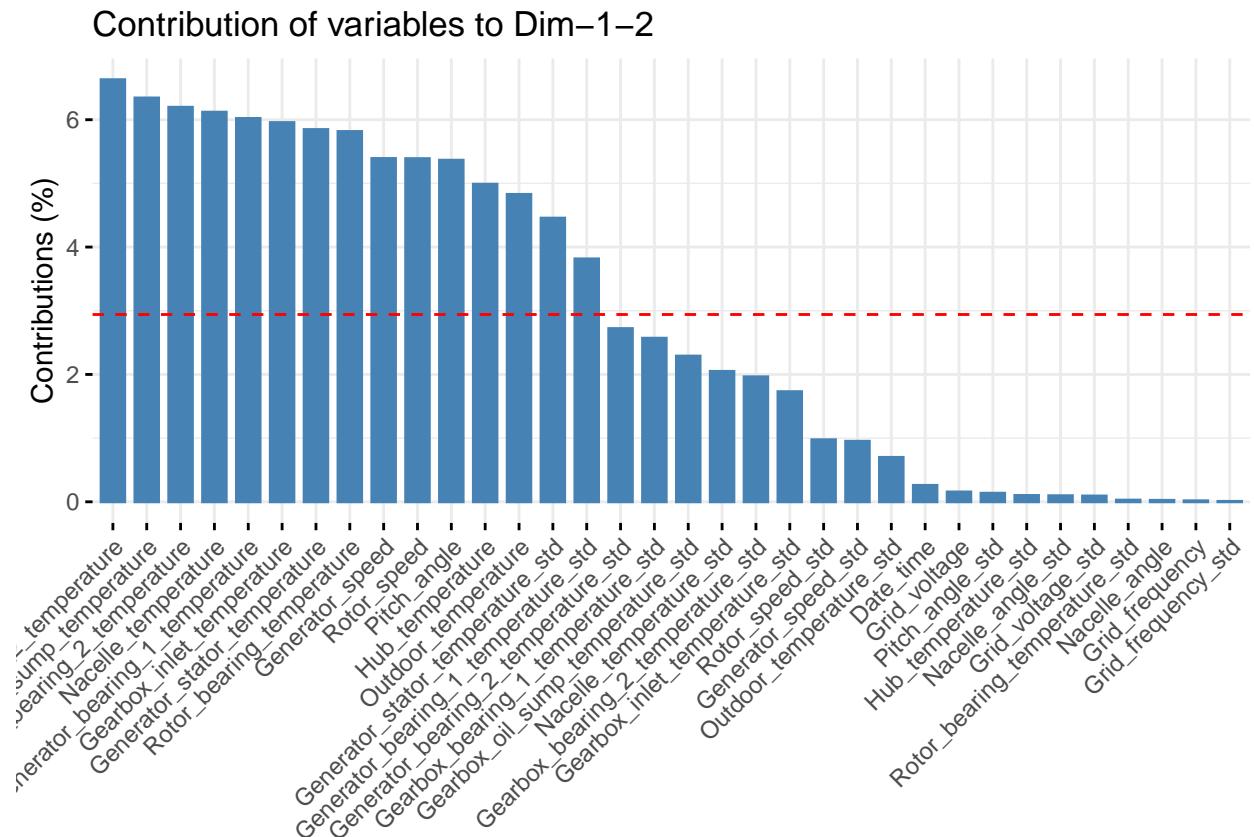
Tableau illustrant une partie de la sortie `dimdesc`

En visualisant les sorties de `dimdesc`, nous observons :

Plusieurs variables sont très liées positivement avec l'axe 1, par exemple :`Generator_speed_std`, `Rotor_speed_std` et `Gearbox_bearing_1_temperature_std`. De plus, nous avons quelques unes liées négativement corrélées, par exemple `Nacelle_angle_std`.

Egalement, nous avons des variables qui sont positivement et négativement avec les axes 2 et 3.

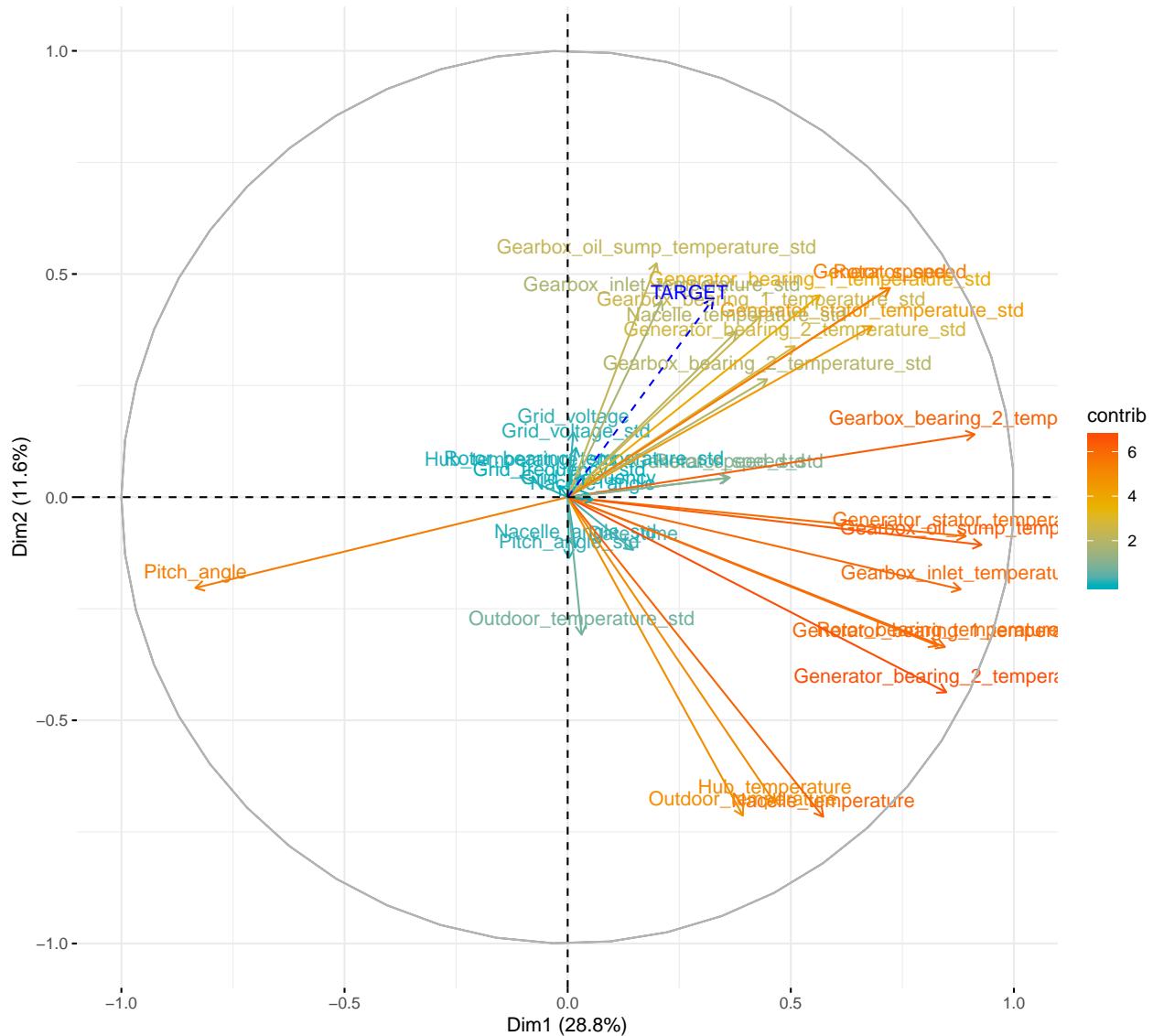
- Contribution des variable à l'axe 1-2



Les variable avec une contribution supérieure à ce seuil représenté en pointillé rouge pourraient être considérées comme importantes pour contribuer à l'axe 1-2, par exemple `Rotor_bearing_temperature`, `Gearbox_inlet_temperature` et `Rotor_speed`.

Nous pouvons également utiliser le cercle de corrélation pour visualiser la contribution des variables à l'axe 1-2.

## Variables – PCA



Le graphe ci-dessus illustre la contribution des variables à l'axe 1-2 selon la couleur, les variables représentées avec la couleur orange sont celle qui contribuent le plus à l'axe 1-2, par exemple `Generator_bearing_2_temperature`.

## Recodage de la variable MAC\_CODE

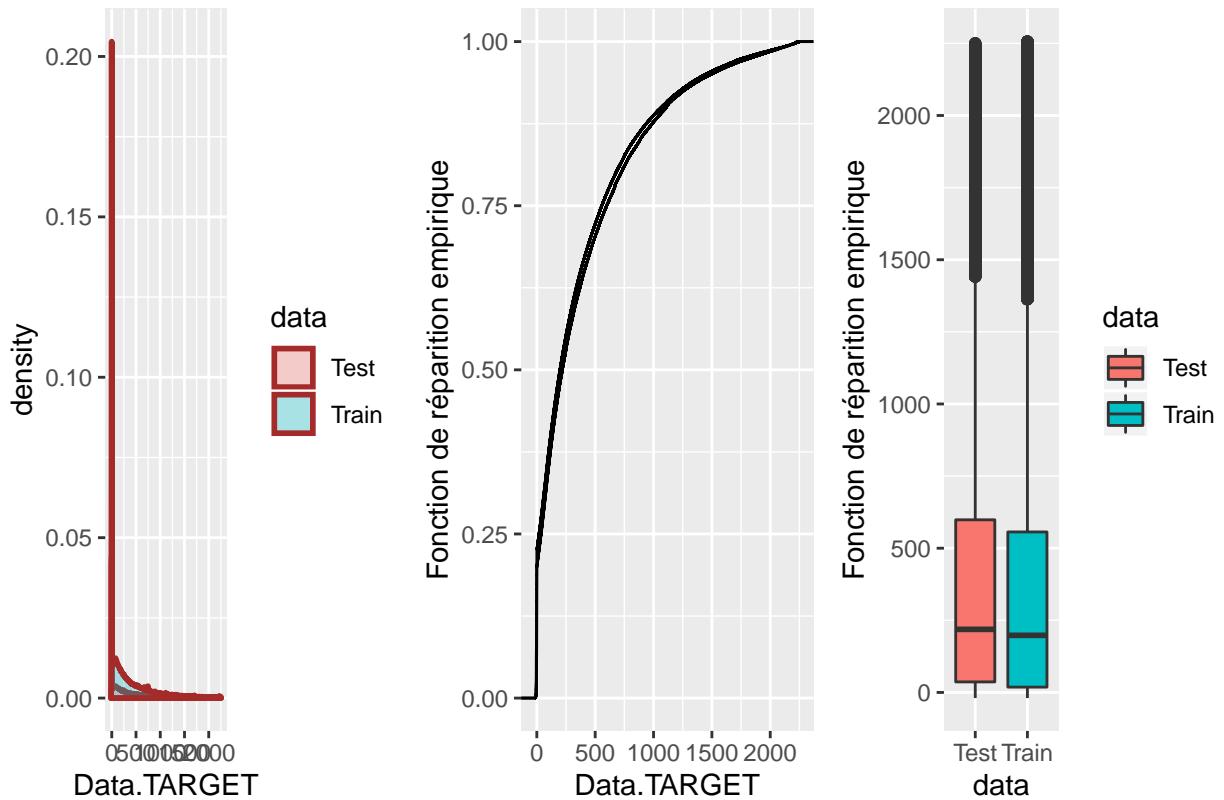
La variable factorielle ‘MAC\_CODE’ a quatre modalité (WT1,WT2,WT3,WT4), nous la recoder de sorte à avoir les modalités {1,2,3,4}.

```
#Régression et sélection de modèles
```

## Erreur de prédiction utilisée

L’erreur de prédiction qui sera utilisée pour toute la suite sur le jeu de données test est définie comme suit : Commençons par appliquer une régression linéaire à notre modèle Complet.

### Comparaison des distributions de la variables TARGET pour train et test



D'après le test de Fisher des hypothèses :

$$H_0 : \beta_1 = \dots = \beta_{35} \text{ vs } H_1 : \exists j \in \{1, \dots, 35\} / \beta_j \neq 0$$

Nous avons la  $p - value < 2.2e - 16$ , nous rejetons donc l'hypothèse  $H_0$  à tout niveau usuel : parmi les 35 variables du modèle, il y'a celles qui sont significatives mais on ne sait pas lesquelles.

Nous pouvons construire un test de niveau  $\alpha = 0.05$  de  $H_0$  vs  $H_1$  en combinant les 35 tests de student :

$$H_0(i) : \beta_i = 0 \text{ vs } H_1(i) \neq 0.$$

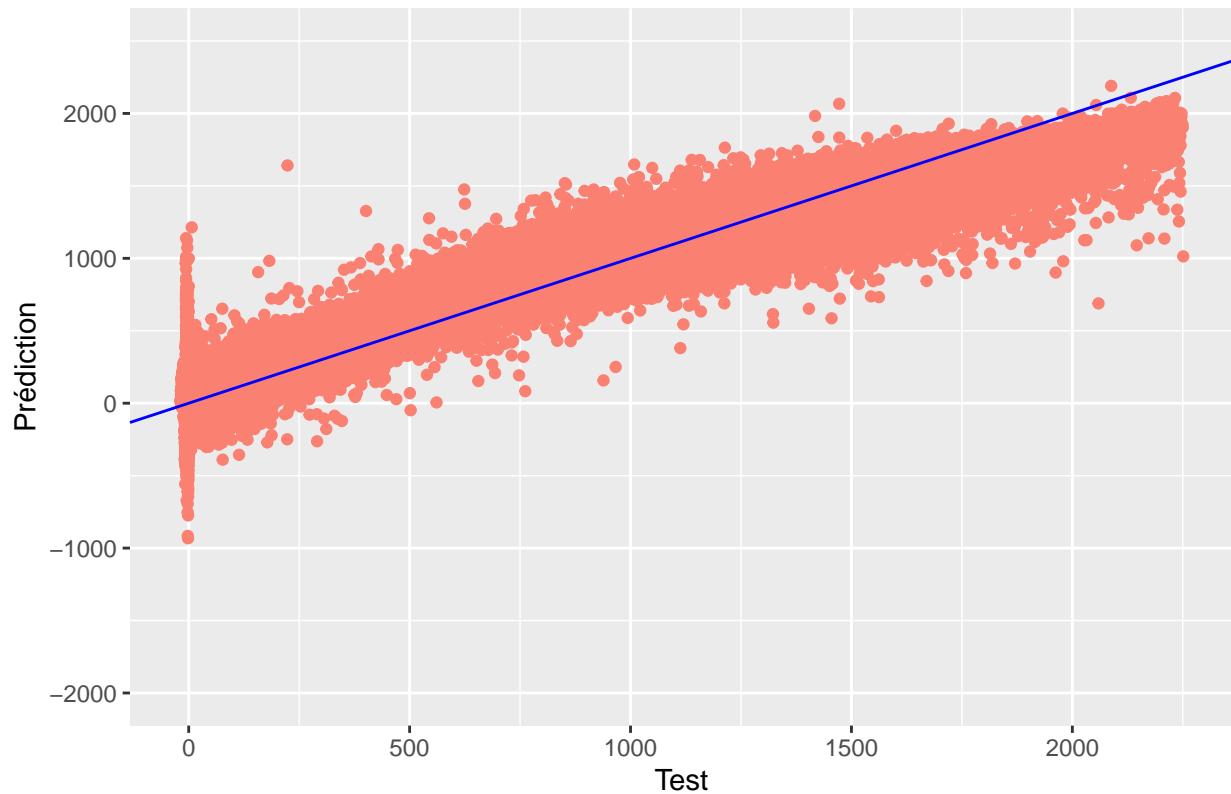
La région de rejet combinée est de la forme :  $R = \bigcup_{i=1}^{35} R_i(\alpha_0)$ .

Le test de student au niveau corrigé  $\alpha_0 = \frac{0.05}{35} = 0.001$ .

D'après les sorties de la fonction `summary(modèle)` toutes les  $p\_value < 0.001$ , nous rejetons donc les hypothèse  $H_0(i) \forall i \in \{1, \dots, 35\}$  et toutes les variables sont significatives.

De plus nous avons  $R^2 = R^2_{adj} = 0.91$ , alors 90% de la variance expliquée de la variable TARGET est expliquée par ce modèle.

## Comparaison prédition et ajustement



L'erreur de prédiction pour ce modèle est égale à **20785.07**.

## Sélection exaustive de variables

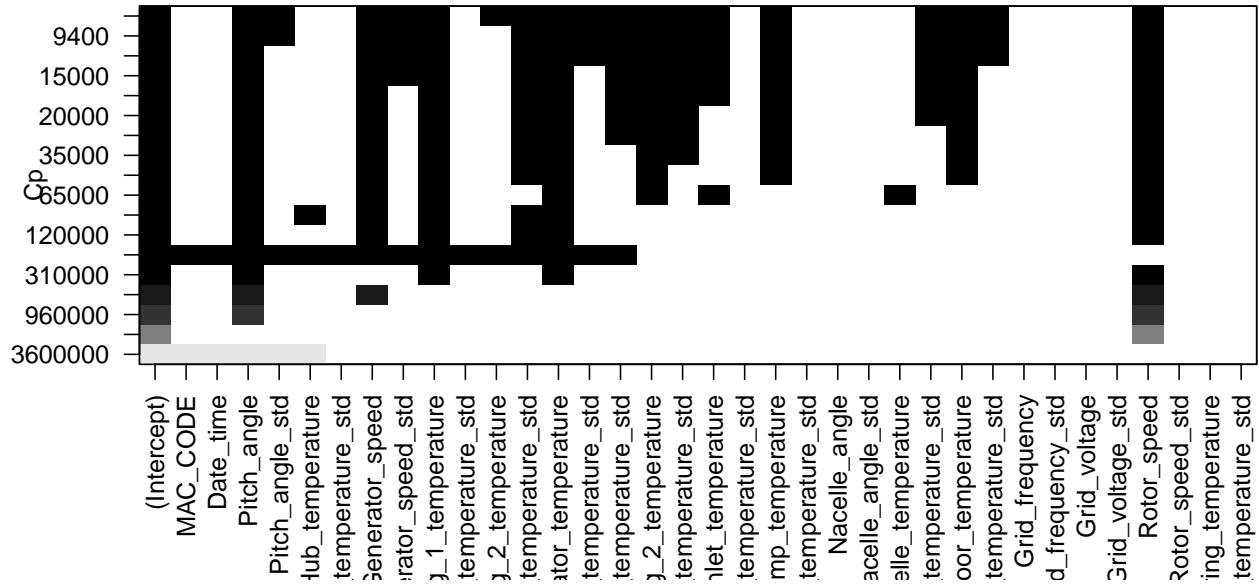
Dans le but d'avoir à la fois le modèle le plus petit possible et qui a une erreur de prédiction minimale, nous allons faire une sélection de variable exaustive en utilisant deux critères :  $C_p$  de Mallows et le *BIC*.

### Critère $C_p$ de Mallows

Le critère 'C\_p\$ de mallows est optimisé pour la ligne en haut du graphique, donc le modèle conservé est à 18 variables (plus la constante) suivant :

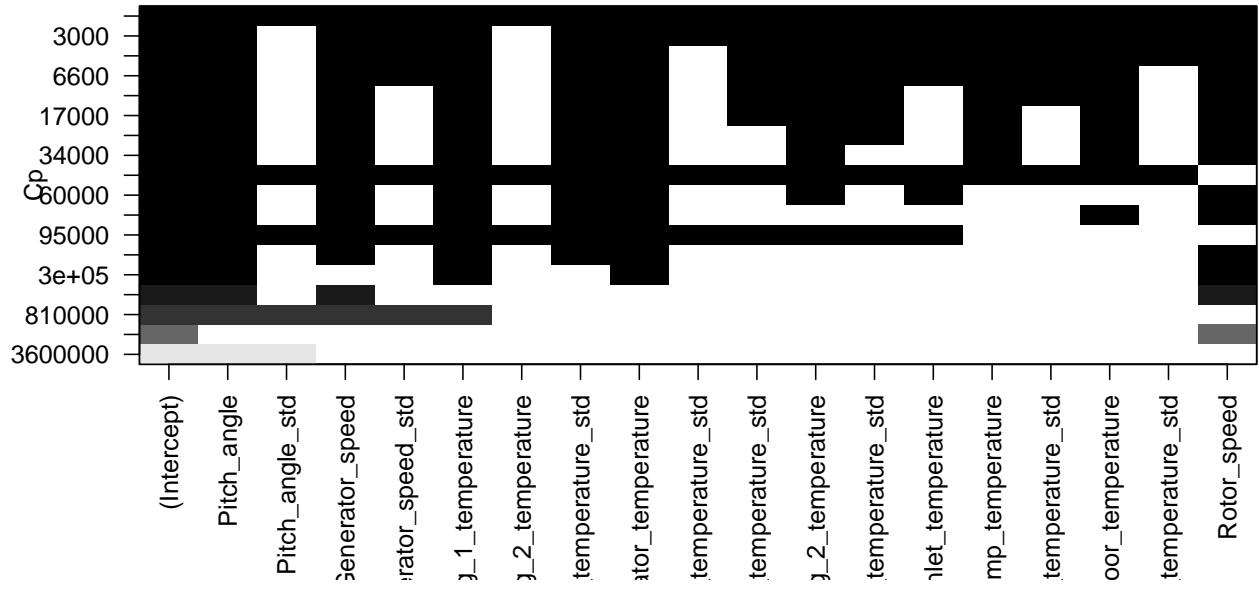
```
TARGET ~ Pitch_angle + Pitch_angle_std + Generator_speed + Generator_speed_std + Generator_bearing_1_temperature + Generator_bearing_2_temperature + Generator_bearing_2_temperature_std + Generator_stator_temperature + Generator_stator_temperature_std + Gearbox_bearing_1_temperature_std + Gearbox_bearing_2_temperature + Gearbox_bearing_2_temperature_std + Gearbox_inlet_temperature + Gearbox_oil_sump_temperature + Nacelle_temperature_std + Outdoor_temperature + Outdoor_temperature_std + Rotor_speed
```

### Cp de Mallows pour le modèle complet



Le critère  $C_p$  de Mallows est optimisé pour la ligne en haut du graphique, nous conservons donc un modèle à 18 variables (plus la constante).

### Cp de Mallows pour le modèle sélectionné

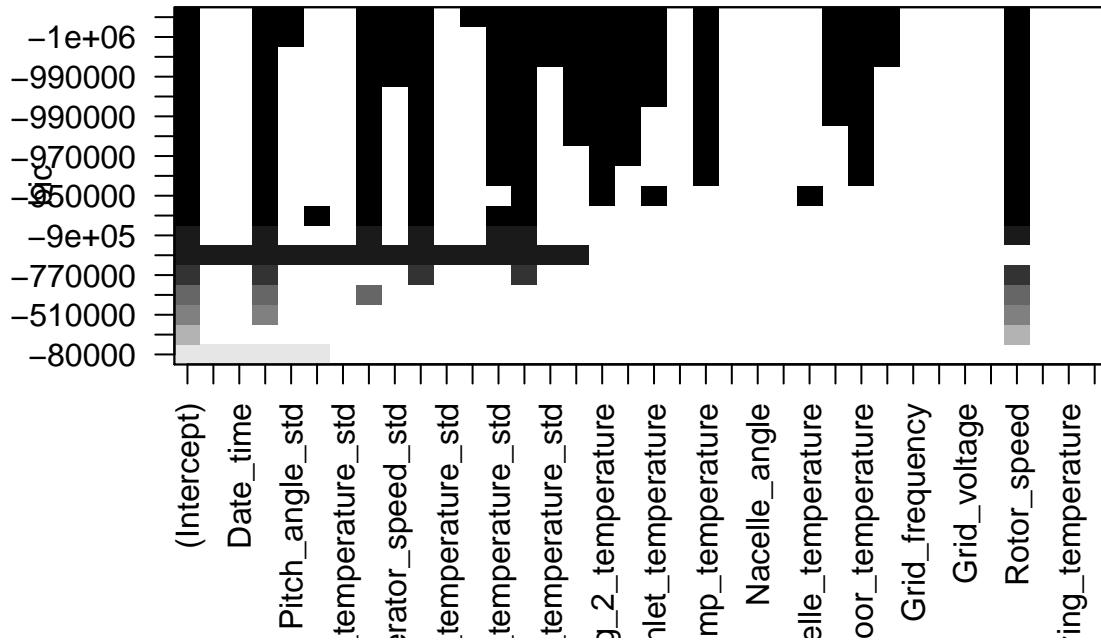


Maintenant, après avoir retracé le plot de sélection de variable sur le nouveau modèle de 18 variables, elles seront toutes conservées.

L'erreur de prédiction correspondant au modèle sélectionné par le  $C_p$  de mallows est de **20190.95** et elle est inférieure à l'erreur de prédiction du modèle complet **20785.07**. Le nouveau modèle explique mieux la variable TARGET.

## Critère BIC

### BIC pour le modèle complet



Le critère BIC est optimisé pour la ligne en haut du graphique, nous conservons donc un modèle à 18 variables (plus la constante) et ce le même modèle retenu avec  $C_p$  de Mallows.

L'erreur de prédiction correspondant au modèle sélectionné par le BIC est de **20190.95** qui est similaire à celle obtenue avec le  $C_p$  de Mallows.

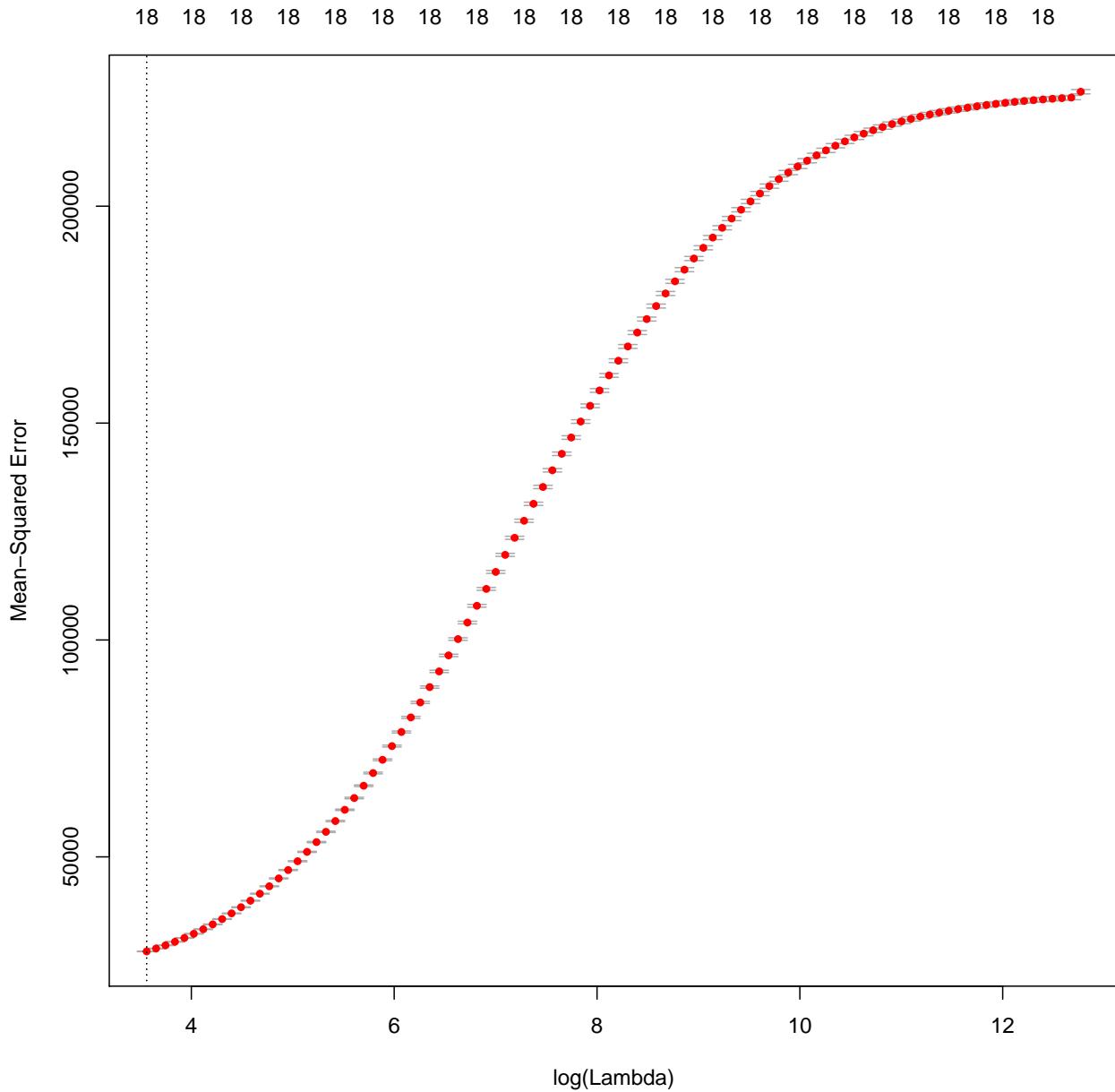
Donc pour la suite, nous allons travailler avec le modèle sélectionné par les deux critères.

## Régression RIDGE

Dans cette partie, nous allons ajuster et prédire avec Ridge. Pour cela, il faut d'abord remplacer les prédicteurs qualitatifs par des "dummy" variables et créer un objet de type matrice contenant les prédicteurs à l'aide de la fonction `model.matrix()`.

On applique la fonction `cv.glmnet` du package `glmnet` qui nous permet de lancer une cross validation sur un modèles RIDGE pour une grille de lambda. Cette fonction permet de sélectionner à partir des données une valeur de lambda qui minimise un critère de validation croisée.

En utilisant la fonction `plot` on obtient directement le graphique représentant des modèles et on extrait celui qui est associé au plus petit lambda.



Par conséquent, la valeur de lambda entraînant la plus petite erreur de validation croisée est 38.56808.

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                                         s0
## Pitch_angle                      4.7089423
## Pitch_angle_std                   4.6136524
## Generator_speed                  0.2758431
## Generator_speed_std              -0.3365382
## Generator_bearing_1_temperature -21.5934090
## Generator_bearing_2_temperature -2.8707309
## Generator_bearing_2_temperature_std -299.5077785
## Generator_stator_temperature    13.1159049
## Generator_stator_temperature_std 5.9459525
## Gearbox_bearing_1_temperature_std 122.6860940
## Gearbox_bearing_2_temperature    14.5425030
## Gearbox_bearing_2_temperature_std -168.0106894
```

```

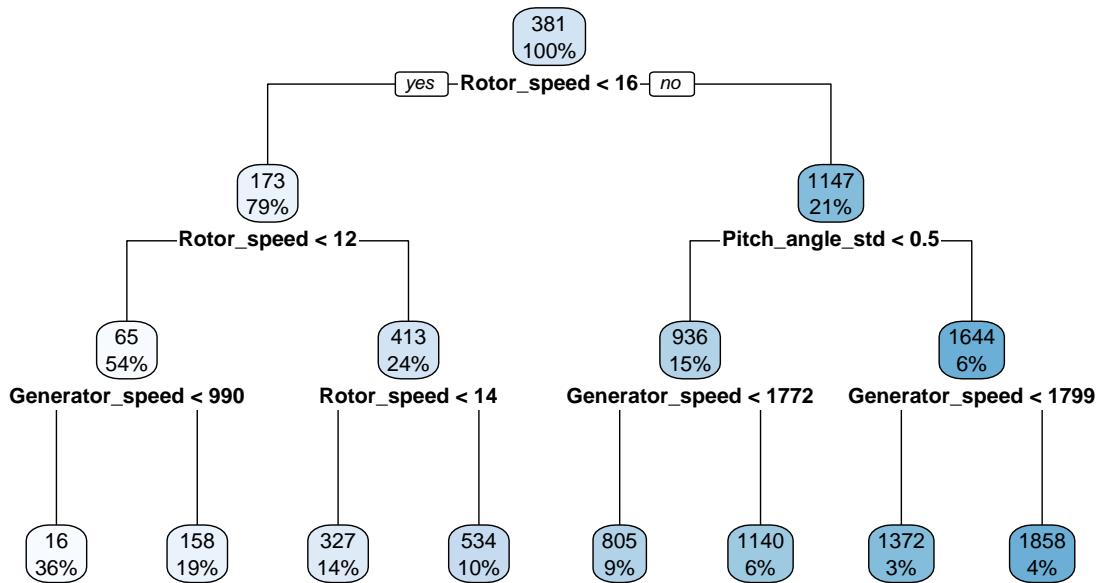
## Gearbox_inlet_temperature      -13.3329586
## Gearbox_oil_sump_temperature   1.7644989
## Nacelle_temperature_std       -30.5889794
## Outdoor_temperature           2.7660319
## Outdoor_temperature_std       147.5024779
## Rotor_speed                   29.1346919

```

Ci-dessous les coefficients  $\beta$  pour  $\lambda = 38.56808$ , aucun des coefficients n'est égale à zéro. Alors la régression de ridge ne permet pas une sélection de variables.

L'erreur de prédiction correspondant à la régression RIDGE est de **30947.94**, elle est supérieure à l'erreur de prédiction du modèle complet **20785.07** et à celle obtenue par le modèle sélectionné par BIC et  $C_p$  de Mallows.

## Arbre de régression



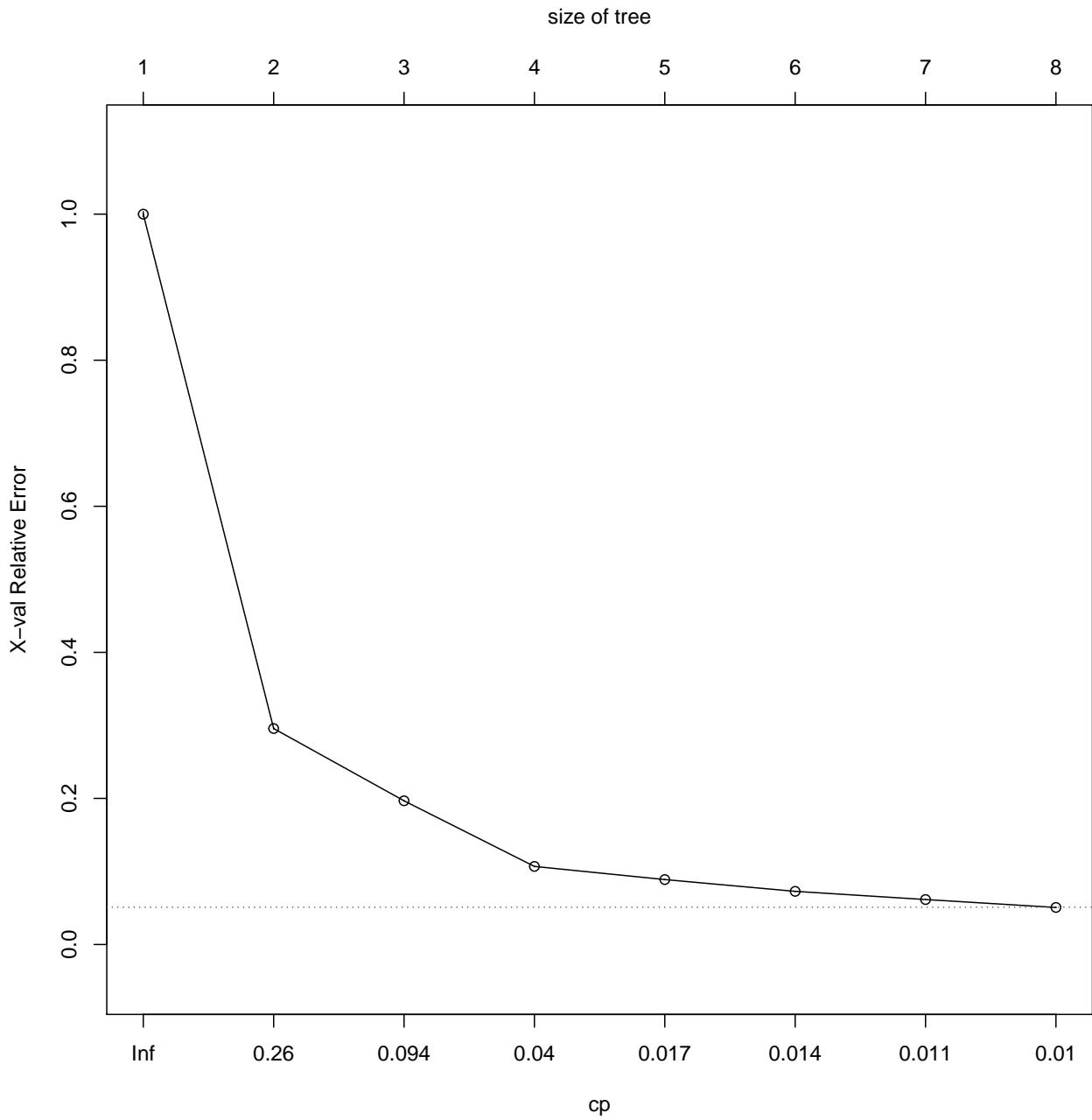
L'arbre présente un noeud qui se divise en deux branches (les branches 2 et 3), en identifiant ceux qui ont des valeurs spécifiques pour la variable `Rotor_speed` (noeuds de gauche nomée `Rotor_speed`). Chaque noeud représente une question, la réponse non étant toujours dans la branche droite de l'arbre.

D'après la sortie de la fonction summary :

- Le critère de split : `Rotor_speed < 11.795` qui correspond au `improve=0.7453816` le plus élevé fait qu'il soit le premier noeud de l'arbre.
- Le nombre total d'instances pour le noeud : 319586.
- La valeur prédictive de la variable à prédire : 172.91070.

La première table du summary (voir aussi cptable ci-dessous) montre comment l'erreur de prédiction (`xerror`) estimée par validation croisée évolue en fonction du paramètre  $C_p$ , ici appelé complexity parameter (`cp` ou `CP`, mais rien à voir avec le `Cp` de Mallows). La colonne `xstd` donne une estimation de l'écart-type de l'erreur de prédiction. La colonne `relative error` évalue l'erreur d'ajustement.

Le summary décrit chaque noeud, on retrouve dans `mean` la valeur de la fonction de régression attachée à la feuille.



`Plotcp()` fournit une représentation graphique au résumé d'erreur validé de manière croisée. Les valeurs de `cp` sont comparées à la moyenne géométrique pour illustrer l'écart jusqu'à ce que la valeur minimale soit atteinte.

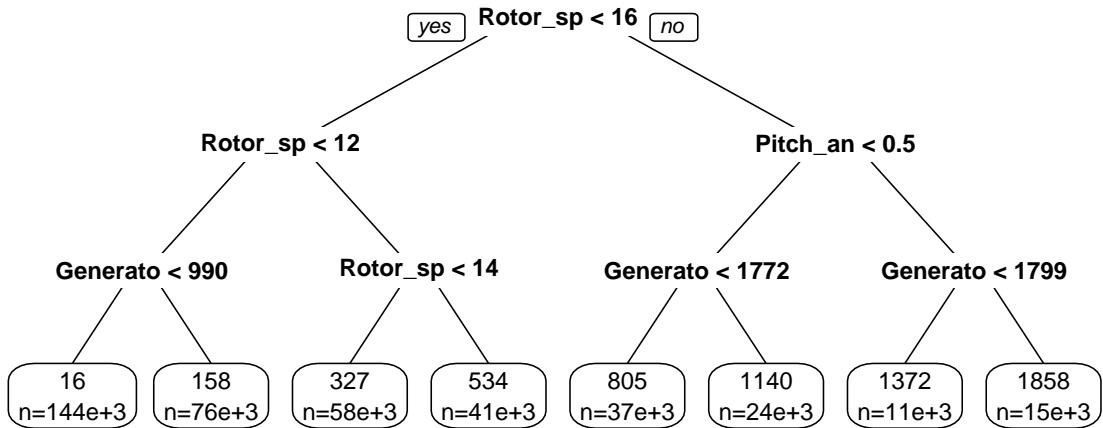
La meilleure erreur de prédiction est obtenue dans la ligne 8. Le `cp` optimal est 0.01.

cp	nsplit	rel error	xerror	xstd
0.0100000000	7.0000000000	0.0505808470	0.0506668353	0.0003283289

On affiche la valeur seuil du paramètre `cp` optimal 0.01.

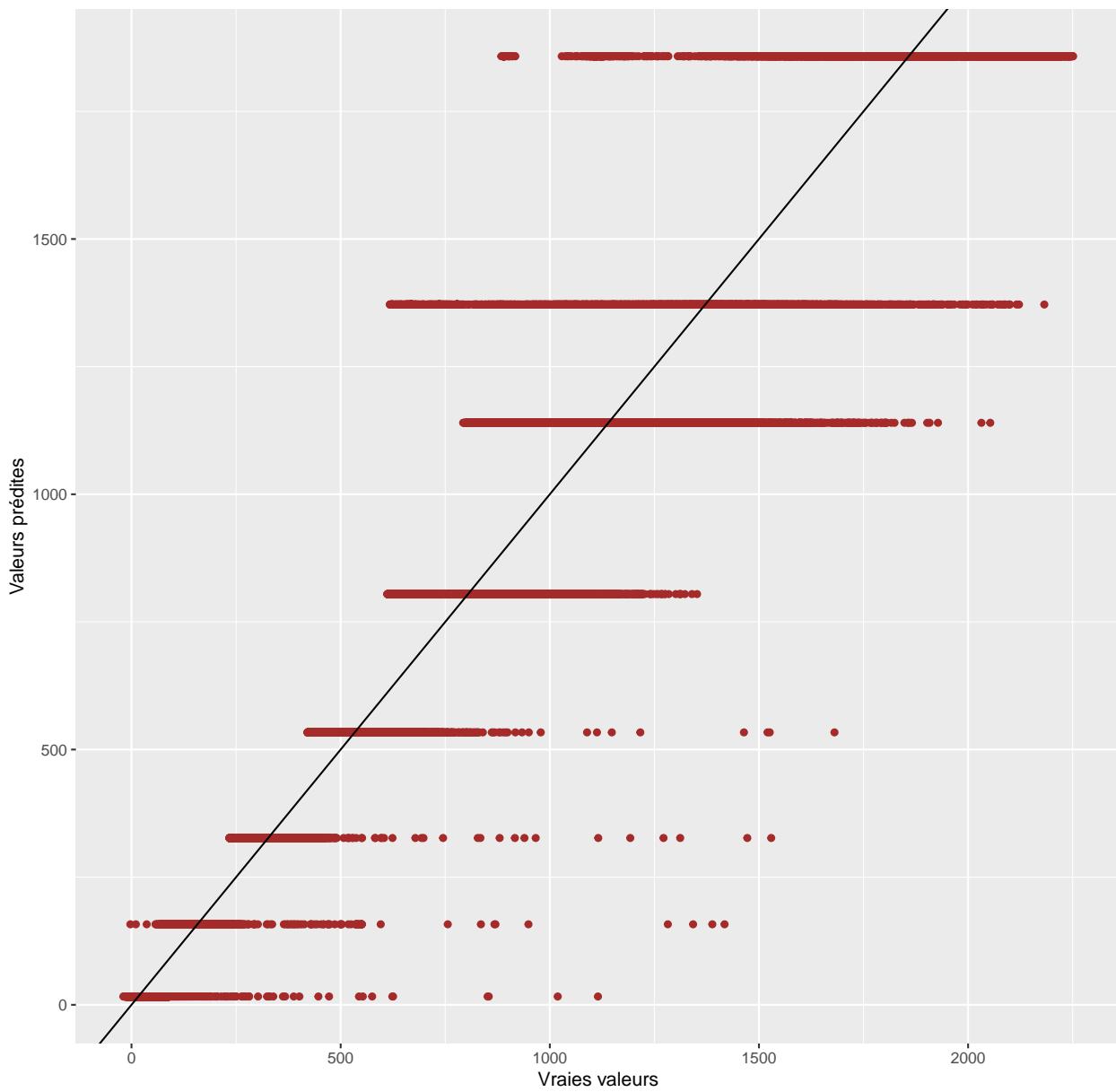
Pour la valeur de `cp` seuil, on a le nombre de splits de l'arbre correspondant (`nsplit`), l'erreur croisée `xerror` et son écart-type `xstd`. On prend en général la première valeur de `cp` (i.e. la plus grande) qui est à moins

de un écart-type du minimum de xerror. Une fois que la valeur de cp est choisie, on peut récupérer l'arbre correspondant par élagage de l'arbre avec le cp optimal 0.01.



Arbre simplifié : chaque nœud représente une question, la réponse non étant toujours dans la branche droite de l'arbre. Chaque feuille est étiquetée par la décision associée et par l'effectif classe par classe des individus affectés à la feuille.

Plot des vraies valeurs et des valeurs prédictes



Sur le graphique ci-dessus qui fait une comparaison entre les valeurs prédictes et les vraies valeurs, on voit que les points sont regroupés en lignes parallèles. En effet, à cause des capacités limitées de notre machine nous n'avons pas pu faire l'arbre maximale, ce qui fait qu'avec notre arbre de régression nous avons peu de feuilles qui ont de grands effectifs (une valeur prédictée peut correspondre à plusieurs valeurs réelles). Donc l'arbre de régression que nous avons construit ne modélise pas bien notre variable à expliquer.

## Conclusion

En conclusion, nous avons vu les étapes qui nous ont donné le meilleur résultat et mené à choisir un modèle. La description des données brutes nous a donné différentes idées. De plus, cela nous a permis de résoudre le problème causé par le dimensionnement de notre jeu de données. Cependant, la régression peut très certainement encore être améliorée en exploitant d'autres pistes, de même pour les résultats de l'arbre de régression en construisant l'arbre maximale. En effet, on peut appliquer d'autres méthodes pour la sélection de variables, on peut également utiliser des algorithmes plus puissants (ex : RandomForest, XGBoost ... etc) et jouer avec leurs paramètres afin d'avoir des modèles plus performants.

```
attach(Data)

Compter <- function(x)
{
  moyenne=mean(x)
  compter1=0
  compter2=0

  for (i in 1:nrow(Data))
  {

    if(x[i]< moyenne)

    {
      compter1=compter1+1
    }
    else
    {
      compter2=compter2+1
    }

    next(i)
  }
  return(compter1,compter2)
}
proportion=c((compter1/(compter1+compter2))*100,(compter2/(compter1+compter2))*100)

}
```