caret

```r
# On télécharge le paquet
install.packages("caret")
```

```r
# on charge le paquet
library(caret)

# On utilisera le dataset 'mtcars'
data("cars")
# Utilisation de 'set.seed()'
set.seed(300)

# On imposera la loi de partition des données

trainIndex <- createDataPartition(cars$Price, p = .8,
                                  list = FALSE,
                                  times = 1)

# On observe le contenu des 6 première ligne du tableau
head(cars)
```

```
##      Price Mileage Cylinder Doors Cruise Sound Leather Buick Cadillac Chevy
## 1 22661.05   20105        6     4      1     0       0     1        0     0
## 2 21725.01   13457        6     2      1     1       0     0        0     1
## 3 29142.71   31655        4     2      1     1       1     0        0     0
## 4 30731.94   22479        4     2      1     0       0     0        0     0
## 5 33358.77   17590        4     2      1     1       1     0        0     0
## 6 30315.17   23635        4     2      1     0       0     0        0     0
##   Pontiac Saab Saturn convertible coupe hatchback sedan wagon
## 1       0    0      0           0     0         0     1     0
## 2       0    0      0           0     1         0     0     0
## 3       0    1      0           1     0         0     0     0
## 4       0    1      0           1     0         0     0     0
## 5       0    1      0           1     0         0     0     0
## 6       0    1      0           1     0         0     0     0
```

```r
# On vérifie le contnue du data set
str(cars)
```

```
## 'data.frame':    804 obs. of  18 variables:
##  $ Price      : num  22661 21725 29143 30732 33359 ...
##  $ Mileage    : int  20105 13457 31655 22479 17590 23635 17381 27558 25049 17319 ...
##  $ Cylinder   : int  6 6 4 4 4 4 4 4 4 4 ...
##  $ Doors      : int  4 2 2 2 2 2 2 2 2 4 ...
##  $ Cruise     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Sound      : int  0 1 1 0 1 0 1 0 0 0 ...
```

```
##  $ Leather    : int  0 0 1 0 1 0 1 1 0 1 ...
##  $ Buick      : int  1 0 0 0 0 0 0 0 0 0 ...
##  $ Cadillac   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Chevy      : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Pontiac    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Saab       : int  0 0 1 1 1 1 1 1 1 1 ...
##  $ Saturn     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ convertible: int  0 0 1 1 1 1 1 1 1 0 ...
##  $ coupe      : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ hatchback  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ sedan      : int  1 0 0 0 0 0 0 0 0 1 ...
##  $ wagon      : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
# On établie les partitions de test et d'entrainement
carsTrain <- cars[trainIndex,]
carsTest <- cars[-trainIndex,]

# zone de test
fitControl <- trainControl(## 5-fold CV
  method = "repeatedcv",
  number = 5,
  # Répeter 5 fois
  repeats = 5)

set.seed(300)

t_before <- Sys.time()
Fit1 <- train(Price ~ ., data = carsTrain,
              method = "rf", # methode d'arbre de décision random forest(rf), on peut tester "nnet" aus
              trControl = fitControl,
              verbose = FALSE)
t_after <- Sys.time()
duree <- t_after - t_before # code pour voir combien de temps ca va durée
print(duree)
```

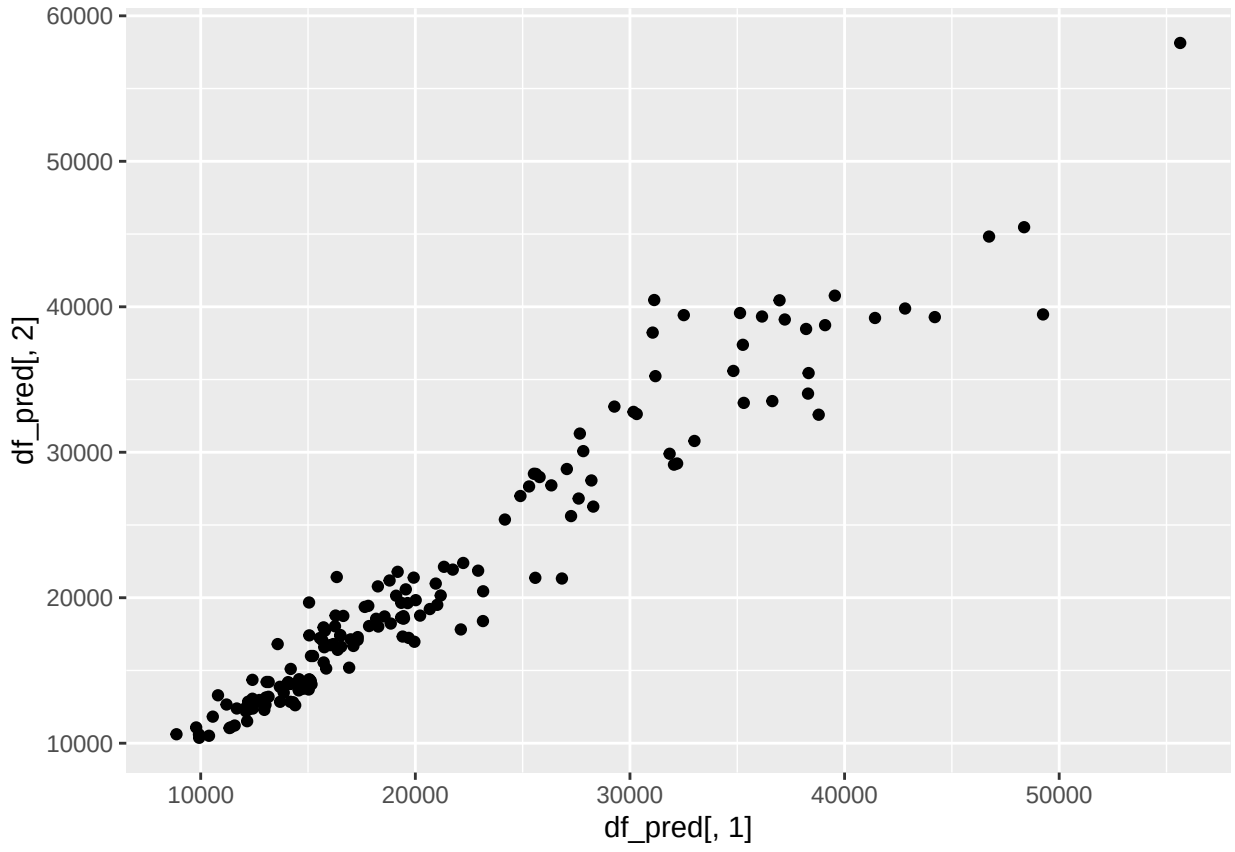
```
## Time difference of 42.23284 secs
```

```
# on teste le modèle
Fit1
```

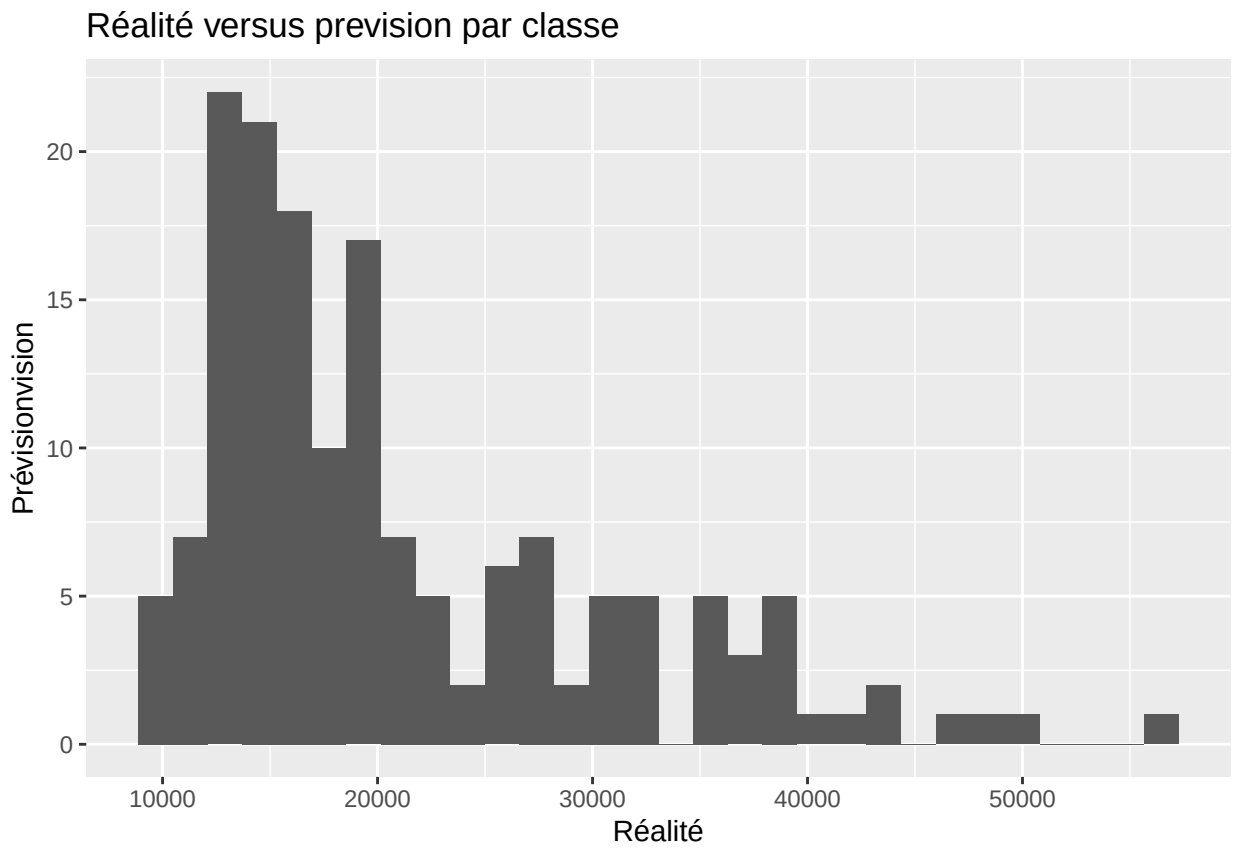
```
## Random Forest
##
## 644 samples
##  17 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 5 times)
## Summary of sample sizes: 516, 514, 515, 516, 515, 516, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared   MAE
##    2    3665.833  0.9116957  2705.964
##    9    2204.804  0.9517320  1538.431
```

```
##   17    2301.991  0.9471867  1620.119
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 9.
```

```r
# Définir le modèle de prédiction pour le test
prediction_carsTest    <- predict(Fit1, carsTest)
df_pred                <- data.frame(carsTest$Price,prediction_carsTest )

ecart <- df_pred[,1] == df_pred[,2]
nombre_reussites <- sum(ecart)
nombre           <- nrow(df_pred)
accuracy         <- nombre_reussites/nombre * 100

# on utilisera le paquet ggplot2
library(ggplot2)

qplot(df_pred[,1],df_pred[,2])
```




```r
# Représentation graphique
ggplot(df_pred) +
  aes(x = carsTest.Price, fill = prediction_carsTest) +
  geom_histogram() +
  scale_fill_hue() +
  labs(x = "Réalité", y = "Prévisionvision", title = "Réalité versus prevision par classe", fill = "Répa
  theme_gray()
```


Réalité versus prevision par classe
Prévisionvision
0
5
10
15
20
10000
20000
30000
40000
50000
Réalité