```
# Illustratie R code voor leren van bomen.
# Het teken “#” wordt gebruikt voor het plaatsen van commentaar zoals op deze regel.
# Commentaar wordt niet geïnterpreteerd door R.

# In R kun je packages (eenmalig) installeren en laden (elke keer dat je de package wilt
gebruiken). De package "farff" laat ons met gemak arff bestanden van Weka in te lezen.

# Laad farff (aannemend dat de package ooit geinstaleerd is)
library(farff)

# Lees de diabetes.arff dataset en bewaar de dataset in "myDataSet".
# Het resultaat "mydataSet" is van het type data.frame.
# Data frames zijn de meest gebruikte form voor datasets.
myDataSet <- readARFF("/Users/ameenabu-
hanna/MIK/Ow/HealthInformatics/BigData/weka-3-8-1/data/diabetes.arff")

# Wat zijn de attributen van myDataSet
names(myDataSet)
```

```
[1] "preg"  "plas"  "pres"  "skin"  "insu"  "mass"  "pedi"  "age"   "class"
```

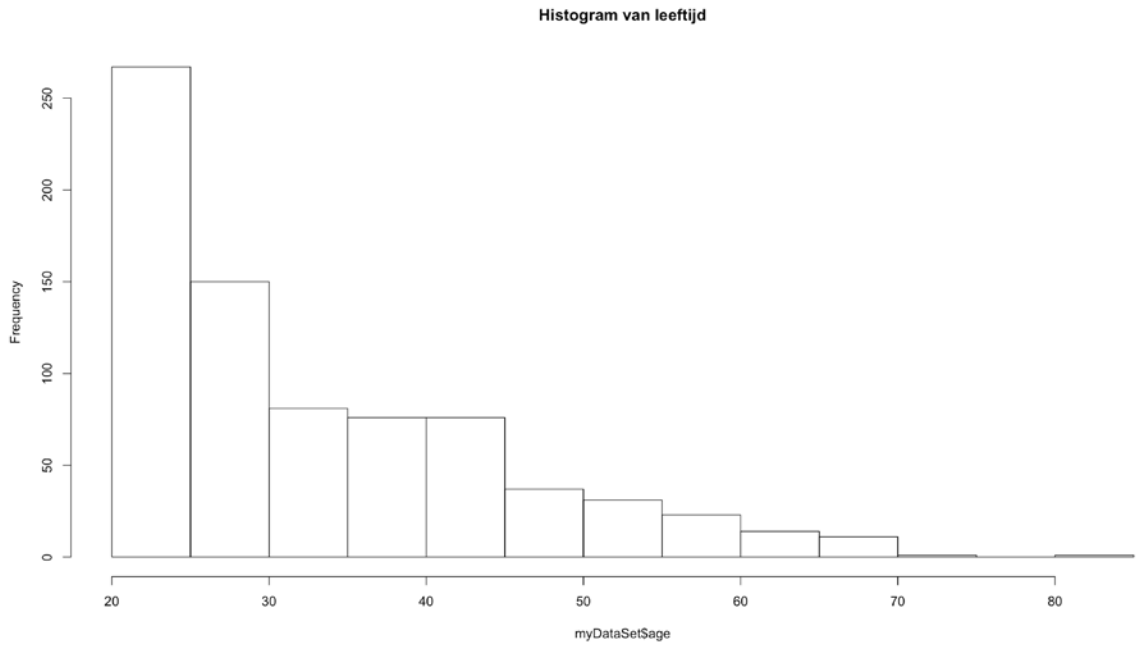
```
# Hoeveel instanties zijn er? (number of rows)
nrow(myDataSet)
```

```
[1] 768
```

```
# Wat zijn de typen van de attributen?
str(myDataSet)
```

```
'data.frame':   768 obs. of  9 variables:
 $ preg : num  6 1 8 1 0 5 3 10 2 8 ...
 $ plas : num  148 85 183 89 137 116 78 115 197 125 ...
 $ pres : num  72 66 64 66 40 74 50 0 70 96 ...
 $ skin : num  35 29 0 23 35 0 32 0 45 0 ...
 $ insu : num  0 0 0 94 168 0 88 0 543 0 ...
 $ mass : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
 $ pedi : num  0.627 0.351 0.672 0.167 2.288 ...
 $ age  : num  50 31 32 21 33 30 26 29 53 54 ...
 $ class: Factor w/ 2 levels "tested_negative",..: 2 1 2 1 2 1 2 1 2 2 ...
```

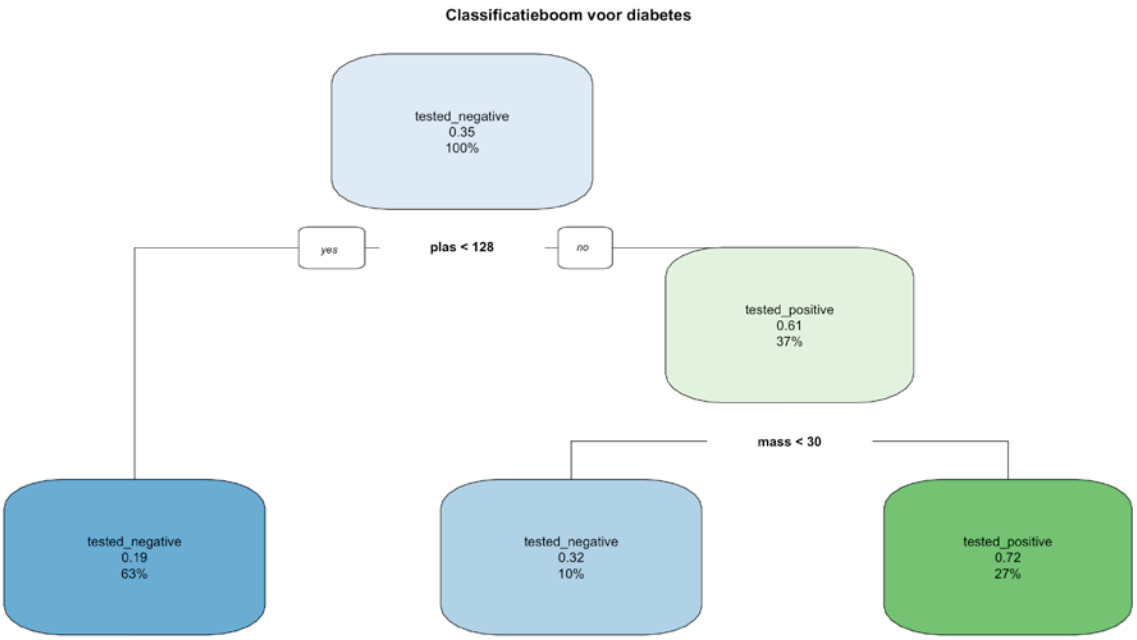
```
# Laten we een histogram maken man leeftijd. Geef titel aan figuur.
hist(myDataSet$age, main="Histogram van leeftijd")
```



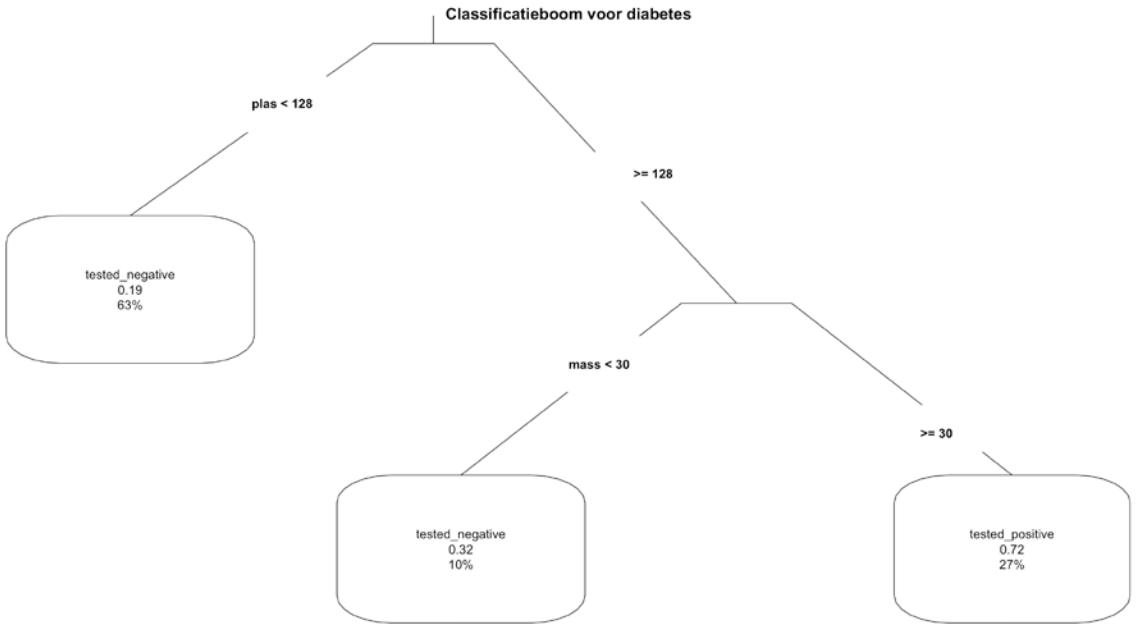

```
# Laten we een 2x2 tabel maken voor het class attribuut
table(myDataSet$class)
```

```
tested_negative tested_positive
            500             268
```

```
# Nu gaan we beslisbomen leren. We gebruiken de package "rpart" (recursive partitioning)
# hiervoor en “rpart.plot” voor het plotten van bomen.

# Laad rpart (aannemend dat de package ooit eerder geïnstalleerd werd)
library(rpart)
# Laad nu rpart.plot (aannemend dat de package ooit eerder geïnstalleerd werd)
library(rpart.plot)

# Leer boom en bewaar het boom object in "mijnboom". Met method ="class" geven we
# aan dat we een classificatieboom willen leren. Met "minsplit = 120" geven we aan dat er
# geen bladeren mogen gegenereerd worden van minder dan 120 instanties. Op deze
# manier zorgen we ervoor dat de boom niet te groot wordt.
mijnboom <- rpart(class ~ preg  + plas  + pres +  skin +  insu  + mass  + pedi +  age, method
="class", minsplit = 120, data=myDataSet)

# plot boom, geef titel aan figuur.
rpart.plot(mijnboom, main="Classificatieboom voor diabetes")
```



# plot boom op andere manier. Zonder kleuren, en de bladeren "hangen" nu
rpart.plot(mijnboom, fallen.leaves=FALSE, type=3, box.palette = 0, main="Classificatieboom voor diabetes")



# Voorspel de uitkomst (tested_negative en tested_positive) op de training set
predictions <- predict(mijnboom, type = "class")

Laat de eerste 10 voorspellingen zien

```
predictions[1:10]
```

```
          1           2           3           4           5
tested_positive tested_negative tested_negative tested_negative tested_positive
          6           7           8           9          10
tested_negative tested_negative tested_negative tested_positive tested_negative
Levels: tested_negative tested_positive
```

```
# Laat de eerste 10 waarden van "class" zien
myDataSet$class[1:10]
```

```
 [1] tested_positive tested_negative tested_positive tested_negative
 [5] tested_positive tested_negative tested_positive tested_negative
 [9] tested_positive tested_positive
Levels: tested_negative tested_positive
```

```
# Vergelijk voorspellingen met de echte class categorieen
table(predictions, myDataSet$class)
```

predictions	tested_negative	tested_positive
tested_negative	443	118
tested_positive	57	150