```
#Parameters Tuning for Decision Tree Model, IST 687 Final Project --sabdelra
```

```
install.packages("caret")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

+ Code + Text

```
install.packages("rio")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependency ‘openxlsx’
```

```
install.packages("mlr")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

```
install.packages("Metrics")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

```
install.packages("rpart")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

```
install.packages("rpart.plot")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

```
install.packages("imputeTS")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

also installing the dependencies ‘xts’, ‘TTR’, ‘markdown’, ‘png’, ‘jpeg’, ‘quadprog’, ‘quantmod’, ‘gridtext’, ‘fracdiff’, ‘lmtest’,
```

```
library (tidyverse)
library(imputeTS)
#library(ggplot)
#library(ggmap)
#library(kernlab)
library(caret)
library(rio)
library(rpart)
library(rpart.plot)
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

```
data <- data.frame(read_csv('HMO_data.csv'))
```

```
Rows: 7582 Columns: 14
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr (8): smoker, location, location_type, education_level, yearly_physical, ...
dbl (6): X, age, bmi, children, hypertension, cost

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
data <- transform(
  data, expensive= ifelse(cost > 5000, TRUE,FALSE))
```

```
data <- data %>% mutate_at(.vars = c("bmi"),
            .funs = ~na_interpolation(.))
```

```
data <- data %>% mutate(across(bmi, ~replace_na(., mean(., na.rm=TRUE))))
```

```
HMO_data <- data[, c('bmi', 'age','smoker', 'exercise', 'expensive')]
```

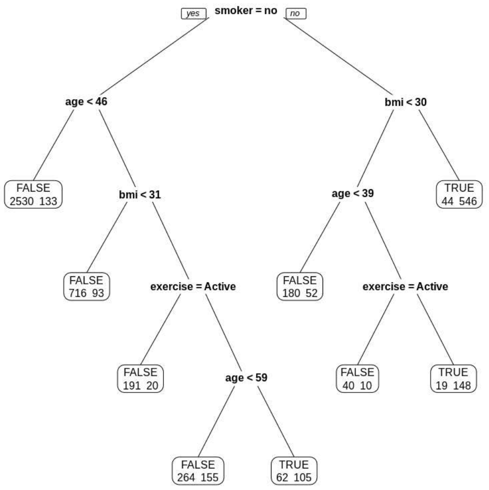
```
HMO_data$expensive <- as.factor(HMO_data$expensive)
```

```
set.seed(111)
trainList <- createDataPartition(y=HMO_data$expensive, p=.70,list=FALSE)
trainSet <- HMO_data[trainList,]
testSet <- HMO_data[-trainList,]
```

```
trainSet$expensive <- as.factor(trainSet$expensive)
trainSet$smoker <- as.factor(trainSet$smoker)
trainSet$exercise <- as.factor(trainSet$exercise)
```

```
cartTree <- rpart(expensive~., data = trainSet,control = c(maxdepth = 5, cp=0.002))
```

```
prp(cartTree, faclen = 0, cex = 0.8, extra = 1)
```



```
predictValues <- predict(cartTree, newdata=testSet, type = "class")
confusionMatrix(predictValues,as.factor(testSet$expensive) )
```

```
Confusion Matrix and Statistics
```

```
library(mlr)
library(Metrics)
```

```
Attaching package: ‘Metrics’


The following objects are masked from ‘package:caret’:

    precision, recall



            Sensitivity : 0.8793
```

```
d.tree.params <- makeClassifTask(
 data=trainSet,
 target="expensive"
 )
```

```
   Detection Prevalence : 0.8333
```

```
param_grid <- makeParamSet(
 makeDiscreteParam("maxdepth", values=1:30))
```

```
# Define Grid
control_grid = makeTuneControlGrid()
# Define Cross Validation
resample = makeResampleDesc("CV", iters = 3L)
# Define Measure
measure = acc
```

```
set.seed(123)
dt_tuneparam <- tuneParams(learner="classif.rpart",
 task=d.tree.params,
 resampling = resample,
 measures = measure,
 par.set=param_grid,
 control=control_grid,
 show.info = TRUE)
```

```
[Tune] Started tuning learner classif.rpart for parameter set:

             Type len Def                                    Constr Req Tunable
maxdepth discrete   -   - 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1...   -    TRUE
         Trafo
maxdepth     -

With control class: TuneControlGrid

Imputation value: -0

[Tune-x] 1: maxdepth=1

[Tune-y] 1: acc.test.mean=0.8513559; time: 0.0 min

[Tune-x] 2: maxdepth=2

[Tune-y] 2: acc.test.mean=0.8524852; time: 0.0 min

[Tune-x] 3: maxdepth=3

[Tune-y] 3: acc.test.mean=0.8749052; time: 0.0 min

[Tune-x] 4: maxdepth=4

[Tune-y] 4: acc.test.mean=0.8830062; time: 0.0 min

[Tune-x] 5: maxdepth=5

[Tune-y] 5: acc.test.mean=0.8830062; time: 0.0 min

[Tune-x] 6: maxdepth=6

[Tune-y] 6: acc.test.mean=0.8830062; time: 0.0 min

[Tune-x] 7: maxdepth=7

[Tune-y] 7: acc.test.mean=0.8830062; time: 0.0 min

[Tune-x] 8: maxdepth=8

[Tune-y] 8: acc.test.mean=0.8830062; time: 0.0 min

[Tune-x] 9: maxdepth=9
```

```
[Tune-y] 9: acc.test.mean=0.8830062; time: 0.0 min

[Tune-x] 10: maxdepth=10

[Tune-y] 10: acc.test.mean=0.8830062; time: 0.0 min

[Tune-x] 11: maxdepth=11

[Tune-y] 11: acc.test.mean=0.8830062; time: 0.0 min

[Tune-x] 12: maxdepth=12

[Tune-y] 12: acc.test.mean=0.8830062; time: 0.0 min
```

```
param_grid_multi <- makeParamSet(
 makeDiscreteParam("maxdepth", values=1:30),
 makeNumericParam("cp", lower = 0.001, upper = 0.01),
 makeDiscreteParam("minsplit", values=1:30)
 )
```

```
dt_tuneparam_multi <- tuneParams(learner="classif.rpart",
 task=d.tree.params,
 resampling = resample,
 measures = measure,
 par.set=param_grid_multi,
 control=control_grid,
 show.info = TRUE)
```

```
[Tune-y] 7820: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7821: maxdepth=21; cp=0.001; minsplit=27

[Tune-y] 7821: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7822: maxdepth=22; cp=0.001; minsplit=27

[Tune-y] 7822: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7823: maxdepth=23; cp=0.001; minsplit=27

[Tune-y] 7823: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7824: maxdepth=24; cp=0.001; minsplit=27

[Tune-y] 7824: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7825: maxdepth=25; cp=0.001; minsplit=27

[Tune-y] 7825: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7826: maxdepth=26; cp=0.001; minsplit=27

[Tune-y] 7826: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7827: maxdepth=27; cp=0.001; minsplit=27

[Tune-y] 7827: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7828: maxdepth=28; cp=0.001; minsplit=27

[Tune-y] 7828: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7829: maxdepth=29; cp=0.001; minsplit=27

[Tune-y] 7829: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7830: maxdepth=30; cp=0.001; minsplit=27

[Tune-y] 7830: acc.test.mean=0.8784850; time: 0.0 min

[Tune-x] 7831: maxdepth=1; cp=0.002; minsplit=27

[Tune-y] 7831: acc.test.mean=0.8513560; time: 0.0 min

[Tune-x] 7832: maxdepth=2; cp=0.002; minsplit=27

[Tune-y] 7832: acc.test.mean=0.8573843; time: 0.0 min

[Tune-x] 7833: maxdepth=3; cp=0.002; minsplit=27

[Tune-y] 7833: acc.test.mean=0.8735857; time: 0.0 min

[Tune-x] 7834: maxdepth=4; cp=0.002; minsplit=27

[Tune-y] 7834: acc.test.mean=0.8830063; time: 0.0 min
```

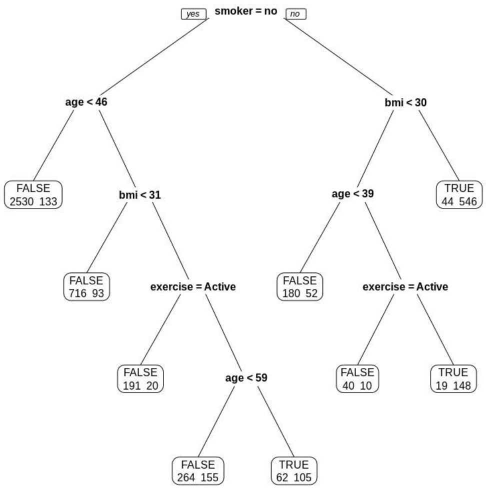
```
best_parameters_multi = setHyperPars(
 makeLearner("classif.rpart", predict.type = "prob"),
 par.vals = dt_tuneparam_multi$x
)

best_parameters_multi
```

```
Learner classif.rpart from package rpart
Type: classif
Name: Decision Tree; Short name: rpart
Class: classif.rpart
Properties: twoclass,multiclass,missings,numerics,factors,ordered,prob,weights,featimp
Predict-Type: prob
Hyperparameters: xval=0,maxdepth=14,cp=0.005,minsplit=5
```

```
cartTree <- rpart(expensive~., data = trainSet,control = c(xval=0,maxdepth=14,cp=0.005,minsplit=5))
```

```
prp(cartTree, faclen = 0, cex = 0.8, extra = 1)
```



```
df <- data.frame(read_csv('HMO_TEST_data_sample.csv'))
df_sol <- data.frame(read_csv('HMO_TEST_data_sample_solution.csv'))
testdf <- df[, c('bmi', 'age','smoker', 'exercise')]
```

```
Rows: 20 Columns: 13
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr (8): smoker, location, location_type, education_level, yearly_physical, ...
dbl (5): X, age, bmi, children, hypertension

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 20 Columns: 2
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
dbl (1): X
lgl (1): expensive

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
predictValues <- predict(cartTree, newdata=testdf, type = "class")
confusionMatrix(predictValues,as.factor(df_sol$expensive) )
```

```
Confusion Matrix and Statistics

          Reference
Prediction FALSE TRUE
     FALSE     9    4
     TRUE      3    4

               Accuracy : 0.65
                 95% CI : (0.4078, 0.8461)
    No Information Rate : 0.6
    P-Value [Acc > NIR] : 0.4159
```

```
predictValues <- predict(cartTree, newdata=testSet, type = "class")
confusionMatrix(predictValues,as.factor(testSet$expensive) )
```

```
Confusion Matrix and Statistics

          Reference
Prediction FALSE TRUE
     FALSE  1686  209
     TRUE     48  331

               Accuracy : 0.887
                 95% CI : (0.8732, 0.8997)
    No Information Rate : 0.7625
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6522

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9723
            Specificity : 0.6130
         Pos Pred Value : 0.8897
         Neg Pred Value : 0.8734
             Prevalence : 0.7625
         Detection Rate : 0.7414
   Detection Prevalence : 0.8333
      Balanced Accuracy : 0.7926

       'Positive' Class : FALSE
```