```r
set.seed(27041970)


source("~/rpart_functions.R")


retention          = read.csv("~/students_retention_full.csv", header = T)
outcome            = c('failed', 'passed')
retention$result   = factor(retention$overall6, levels = 0:1, labels = outcome)
retention$overall6 = NULL
retention$id       = NULL

attach(retention)
table(retention$result)

include    = rbinom(dim(retention)[1], 1, 2/3)
training   = subset(retention, include == 1)
validation = subset(retention, include == 0)



fit0 = fit.buildP(result ~ ., data = training)
fit.plot(fit0)
fit.plotcp(fit0)

prediction = fit.predict(fit0, validation)
fit.performance(fit0, prediction, validation, rev(outcome))
fit.analysis(fit0, prediction, validation, rev(outcome))


fit1 = fit.buildX(result ~ ., data = training)
fit.plot(fit1)
fit.plotcp(fit1)

prediction = fit.predict(fit1, validation)
fit.performance(fit1, prediction, validation, rev(outcome))
fit.analysis(fit1, prediction, validation, rev(outcome))


fit2 = fit.prune(fit1, cp = 0.0095785)
fit.plot(fit2)
fit.plotcp(fit2)

prediction = fit.predict(fit2, validation)
fit.performance(fit2, prediction, validation, rev(outcome))
fit.analysis(fit2, prediction, validation, rev(outcome))


fit3 = fit.prune(fit1, cp = 0.0114943)
fit.plot(fit3)
fit.plotcp(fit3)

prediction = fit.predict(fit3, validation)
fit.performance(fit3, prediction, validation, rev(outcome))
fit.analysis(fit3, prediction, validation, rev(outcome))


fit4 = fit.prune(fit1, cp = 0.0517241)
fit.plot(fit4)
fit.plotcp(fit4)

prediction = fit.predict(fit4, validation)
fit.performance(fit4, prediction, validation, rev(outcome))
fit.analysis(fit4, prediction, validation, rev(outcome))


fit5 = fit.prune(fit1, cp = fit.mincvcp(fit1))
fit.plot(fit5)
fit.plotcp(fit5)

prediction = fit.predict(fit5, validation)
fit.performance(fit5, prediction, validation, rev(outcome))
fit.analysis(fit5, prediction, validation, rev(outcome))
```

```r
# for the specific example of lcpoints we will find the splitting point using gini scoring.

score  = function(p1, p2) { 1 - p1^2 - p2^2 }
# score = function(p1, p2) { -((p1 * log(p1)) + (p2 * log(p2))) }

calculate = function(split1, split2) {

  length1 = nrow(split1)
  length2 = nrow(split2)
  prob1   = prop.table(table(split1$result))
  prob2   = prop.table(table(split2$result))

  (length1 / length) * score(prob1[1], prob1[2]) + (length2 / length) * score(prob2[1], prob2[2])
}

length  = nrow(training)
points  = sort(unique(training$lcpoints))
minimum = c(1.0, 0)

for (point in points) {

  value = calculate(training[ training$lcpoints  < point, ], training[ training$lcpoints >= point, ])
  if (is.finite(value) && value < minimum[1])
    minimum = c(value, point)
}

minimum









detach(retention)
```