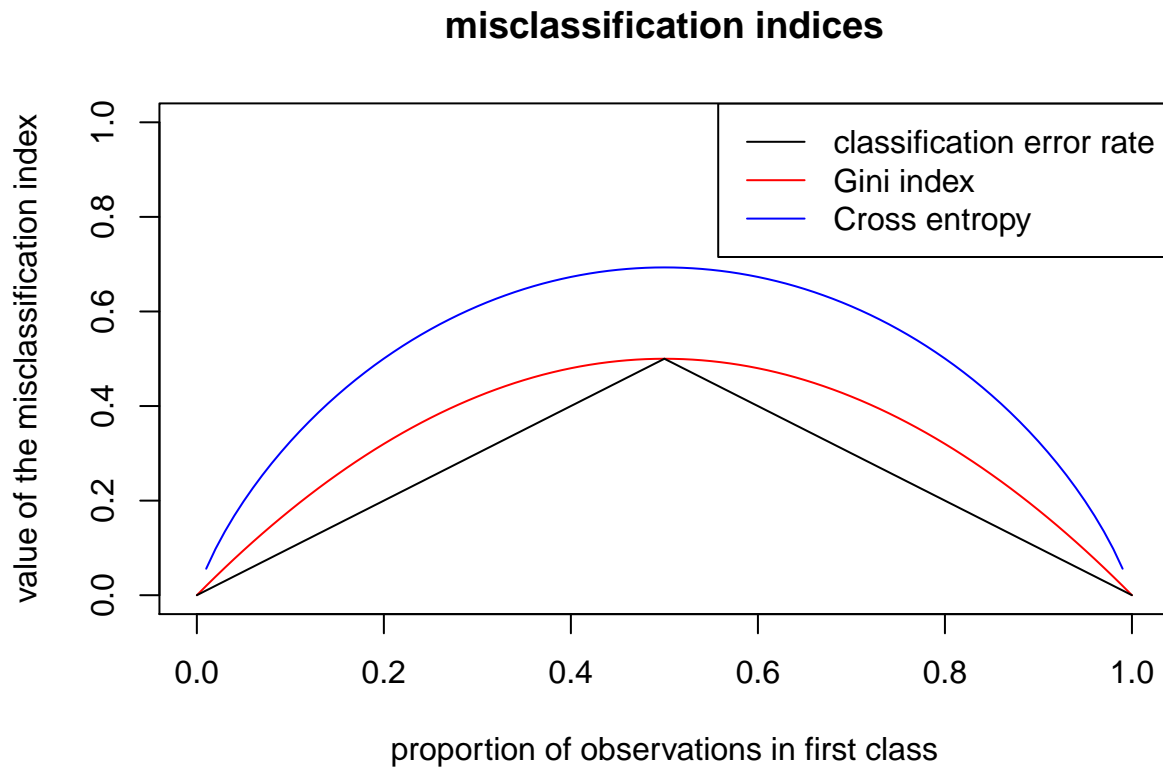


Exercises IOPS course SLP: Trees and ensembles

Exercise 1: Comparing misclassification indices



The plot above, show the values of the different misclassification indices, given the proportion of class-1 observations in a node, on the x-axis.

- Based on the plot, do you expect each of the criteria to favor the same, or different potential splits?
- Say, we have a mothernode with .7999 in class 1 (and .2111 in class 0). A given split would result in
 - .565 going left, of which a proportion of 1.00 are class 1 observations
 - .435 going right, of which a proportion of .54 are class 1 observations

Calculate the (average of the) classification error, Gini index and cross-entropy in the mothernode, and in the two daughternodes. Would the split improve purity according to the classification error, Gini index and cross-entropy?

Classification error in the mothernode is

```
1 - .7999
```

```
## [1] 0.2001
```

Classification in the daughter nodes will be

```
.565 * (1 - 1.0) + .435 * (1 - .54)
```

```
## [1] 0.2001
```

Gini index in the mothernode is

```
.7999 * .2001 + .2001 * .7999
```

```
## [1] 0.32012
```

Gini index in the daughternodes will be

```
.565 * (1.0 * 0.0 + 0.0 * 1.0) + .435 * (.54 * .46 + .46 * .54)
```

```
## [1] 0.216108
```

Cross-entropy in the mothernode is

```
- (.7999 * log(.7999) + .2001 * log(.2001))
```

```
## [1] 0.500531
```

Cross-entropy in the daughternodes will be (have to pick a very small value instead of zero, otherwise log is not defined)

```
- (.565 * (1.00 * log(1.00) + 0.00 * log(1e-50)) +  
.435 * (.54 * log(.54) + .46 * log(.46)))
```

```
## [1] 0.3001255
```

Conclusion: According to the classification error rate, purity would not improve and no split would be made. According to the Gini index and cross-entropy, purity would improve and a split would be made.

Exercise 2: Variable selection bias

- a) Set the random seed and generate 200 observations from independent variables x_1 , x_2 and e (you are free to choose the shape and parameters of the distribution yourself). Create two datasets consisting of x_1 , x_2 and y : one where $y = e$ (the 'independent' dataset), and one where $y = x_2 + e$ (the 'dependent' dataset).

```
set.seed(42)  
x1 <- rnorm(200)  
x2 <- round(x1)  
e <- rnorm(200)  
indep.data <- data.frame(x1, x2, y = e)  
dep.data <- data.frame(x1, x2, y = x2 + e)  
cor(indep.data)
```

```
##           x1           x2           y  
## x1  1.00000000  0.95421214 -0.08036539  
## x2  0.95421214  1.00000000 -0.08438802  
## y   -0.08036539 -0.08438802  1.00000000
```

```
cor(dep.data)
```

```
##           x1           x2           y  
## x1  1.00000000  0.9542121  0.6653023  
## x2  0.9542121  1.0000000  0.6971071  
## y   0.6653023  0.6971071  1.0000000
```

- b) Fit a regression tree using x_1 and x_2 to predict y , using each dataset. Which variable is most often selected for splitting? Is that what you would expect?

```
library(tree)
indep.tree <- tree(y ~ ., data = indep.data)
indep.tree
```

```
## node), split, n, deviance, yval
##      * denotes terminal node
##
##  1) root 200 178.5000  0.01128
##    2) x1 < 0.908147 167 127.8000  0.07600
##      4) x1 < 0.702793 159 123.4000  0.04904
##        8) x1 < 0.642954 152 116.5000  0.08731
##          16) x1 < 0.529576 141 110.0000  0.05215
##            32) x1 < 0.443734 135 105.4000  0.08183
##              64) x1 < -0.611299 50  32.0400  0.24060
##                128) x1 < -1.06408 29  16.2900 -0.05572 *
##                129) x1 > -1.06408 21   9.6790  0.64990 *
##              65) x1 > -0.611299 85  71.3600 -0.01157
##                130) x1 < -0.432308 15   6.5260 -0.43390 *
##                131) x1 > -0.432308 70  61.5900  0.07892 *
##            33) x1 > 0.443734 6   1.7880 -0.61570 *
##          17) x1 > 0.529576 11   4.1010  0.53800 *
##        9) x1 > 0.642954 7    1.8410 -0.78210 *
##      5) x1 > 0.702793 8    2.0080  0.61190 *
##    3) x1 > 0.908147 33  46.4400 -0.31620
##      6) x1 < 1.83186 28  38.2500 -0.46980
##        12) x1 < 1.06526 5   7.8830 -1.08900 *
##        13) x1 > 1.06526 23  28.0300 -0.33510
##          26) x1 < 1.20123 5   0.8233  0.45050 *
##          27) x1 > 1.20123 18  23.2600 -0.55340
##            54) x1 < 1.38103 7   5.7970 -1.05300 *
##            55) x1 > 1.38103 11  14.6100 -0.23540 *
##      7) x1 > 1.83186 5   3.8390  0.54360 *
```

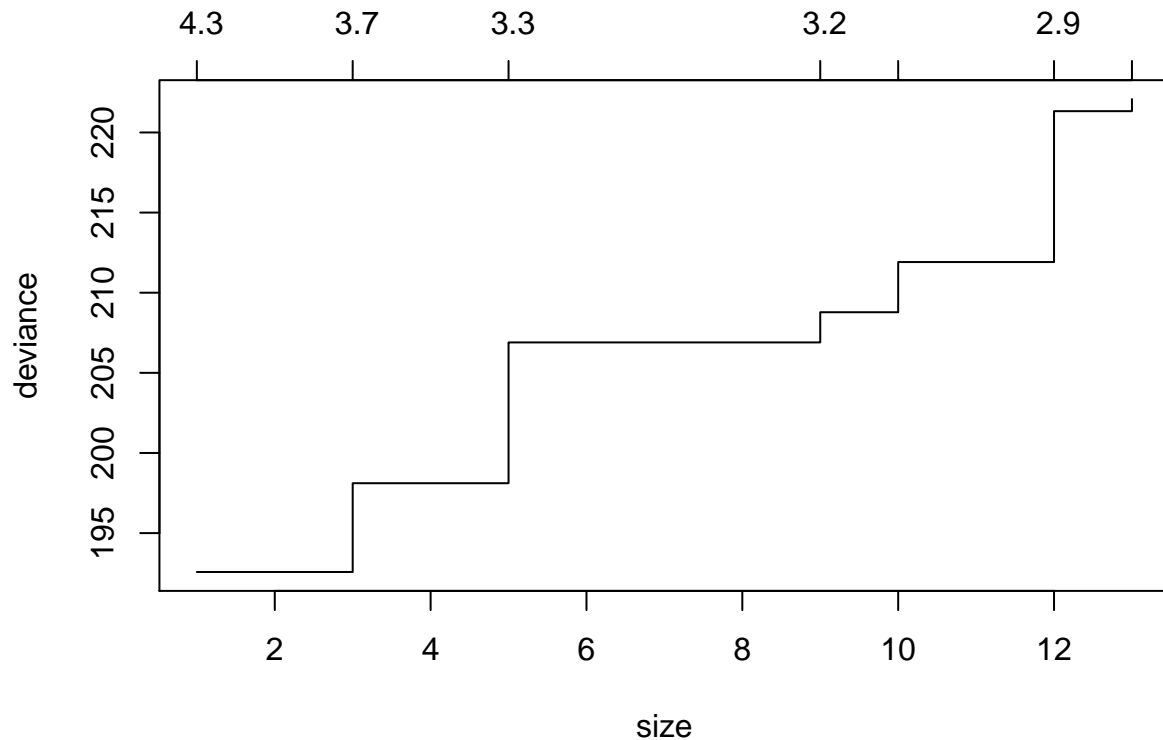
```
dep.tree <- tree(y ~ ., data = dep.data)
dep.tree
```

```
## node), split, n, deviance, yval
##      * denotes terminal node
##
##  1) root 200 344.800  0.011280
##    2) x1 < 0.494074 140 163.800 -0.452700
##      4) x1 < -1.06408 29  22.610 -1.538000
##        8) x1 < -1.52258 12   7.732 -2.065000 *
##        9) x1 > -1.52258 17   9.208 -1.167000 *
##      5) x1 > -1.06408 111  98.070 -0.169000
##        10) x1 < -0.483214 30  17.680 -0.637500
##          20) x1 < -0.611299 21   9.679 -0.350100 *
##          21) x1 > -0.611299 9   2.221 -1.308000 *
##        11) x1 > -0.483214 81  71.360  0.004492 *
##    3) x1 > 0.494074 60  80.500  1.094000
##      6) x1 < 1.83186 55  61.780  0.943900
##        12) x1 < 0.634407 10   4.381  1.542000 *
##        13) x1 > 0.634407 45  53.020  0.811000 *
##      7) x1 > 1.83186 5   3.879  2.744000 *
```

Both the independence and dependence tree use only x_1 for splitting. Taking into account the population and sample correlations between x_1 , x_2 and y , one would expect that x_2 would be selected at least as often as x_1 for splitting.

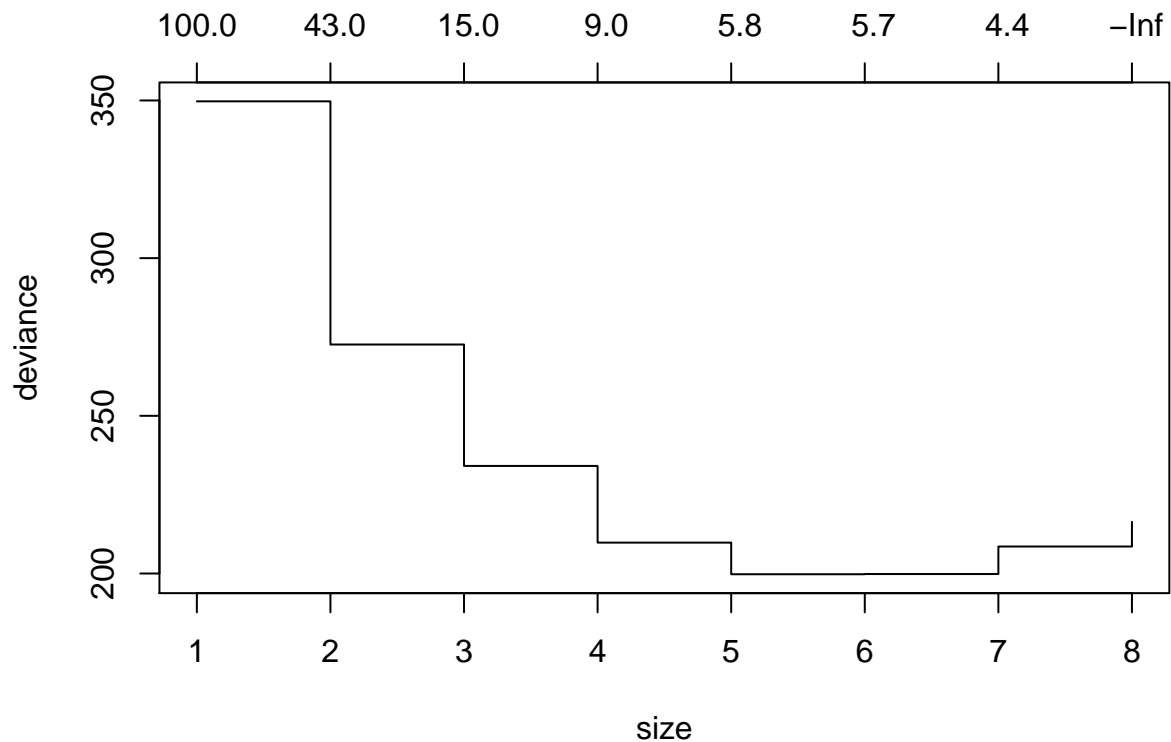
c) Prune the trees. Are there any splits left?

```
set.seed(4287243)
cv.indeptree <- cv.tree(indep.tree)
plot(cv.indeptree)
```



The independence tree should be pruned to a size-1 tree, so there are no splits left.

```
set.seed(4287243)
cv.deptree <- cv.tree(dep.tree)
plot(cv.deptree)
```

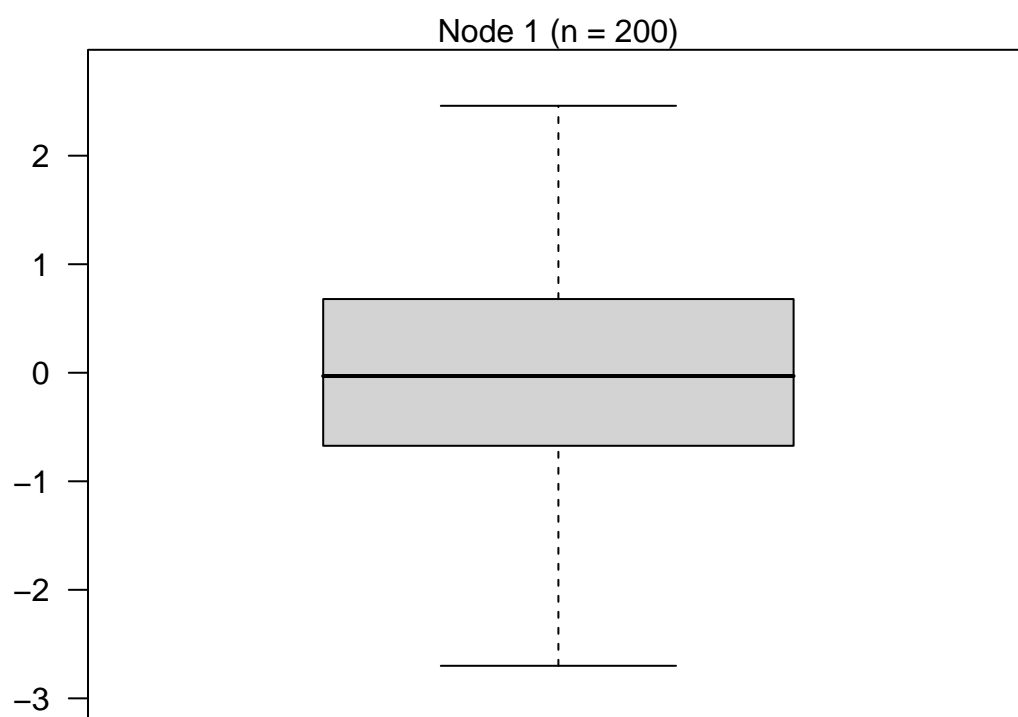


The dependence tree should be pruned to 5 terminal nodes.

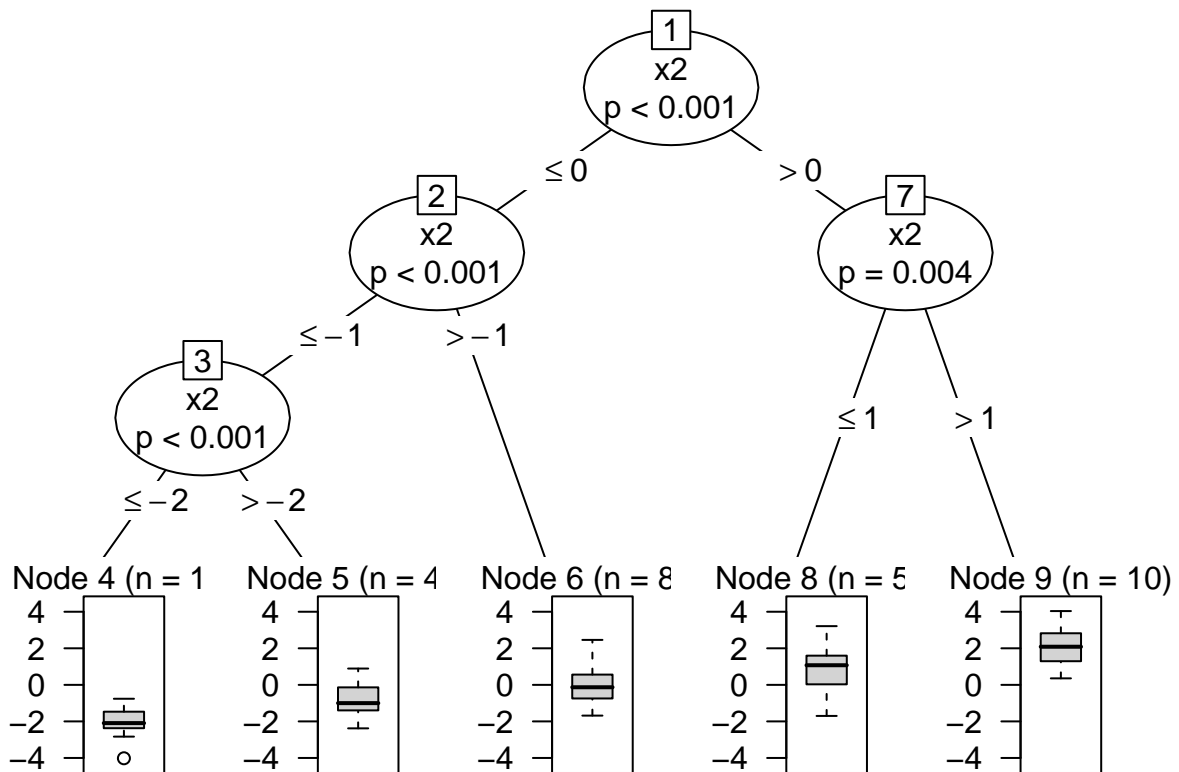
- c) Use the `ctree()` function from the `partykit` package to fit a conditional inference tree to the independent and dependent data. Also plot the resulting conditional inference tree.

```
library(partykit)

## Loading required package: grid
## Loading required package: libcoin
## Loading required package: mvtnorm
## Loading required package: rpart
indep.ctree <- ctree(y ~ ., data = indep.data)
plot(indep.ctree)
```



```
dep.ctree <- ctree(y ~ ., data = dep.data)
plot(dep.ctree)
```



d) Compare the results you obtained in part a, b and c.

The conditional inference tree has a preference for x_2 , as it is slightly higher correlated with y . The CART tree has a preference for x_1 , as it has more potential splitting values than x_2 . The conditional inference tree does not need to be pruned, as it quits splitting when there is no significant association between the predictor and outcome variables anymore.

Exercise 3: Fitting trees and ensembles to the Carseats data

(Adaptation of exercise 8.8 ISLR)

In the lab session of chapter 8 (ISLR), a classification tree was applied to the Carseats data set after converting Sales into a qualitative response variable. Now we will seek to predict Sales using regression trees and related approaches, treating the response as a quantitative variable.

- Split the data set into a training set and a test set.
- Fit a regression tree to the training set. Plot the tree, and interpret the results. What test MSE do you obtain?
- Use cross-validation in order to determine the optimal level of tree complexity. Does pruning the tree improve the test MSE?
- Use the bagging approach in order to analyze this data. What test MSE do you obtain? Use the `importance()` function to determine which variables are most important.
- Fit a random forest:

- e) Use random forests to analyze this data. What test MSE do you obtain? Use the `importance()` function to determine which variables are most important. Describe the effect of `mtry`, the number of variables considered at each split, on the error rate obtained.
- f) Create a boosted ensemble to predict Sales. Compare the boosted ensemble with the bagged and random forest ensemble in terms of test MSE and (the effect of) important predictor variables. (Additional: Before creating the ensemble, use cross validation to determine the optimal parameter settings.)

Exercise 4: Boston housing and OOB error estimates

(Adaptation of exercise 8.7 ISLR).

In the lab, a random forest was created for the Boston data using `mtry=6` and using `ntree=25` and `ntree=500`. For `mtry` values of p , $p/2$, and \sqrt{p} . Use `ntree` values of 1:750. Create a plot with the number of trees on the x-axis and the error rate on the y-axis. Plot both the OOB and test error.

Hints: Note that you only need to fit 3 ensembles, one for each value of `mtry`, because the fitted `randomForest` object contains a slot `$mse`, of which the i -th element ($1 \leq i \leq ntree$) is the OOB estimate of the MSE for all trees up to the i -th; and a slot `$testmse`, of which the i -th element ($1 \leq i \leq ntree$) is the test MSE for the ensemble of trees up to the i -th.

To obtain both OOB and test error, first separate the data in a test and training set and supply these to the `X.train`, `Y.train`, `xtest` and `ytest` arguments of the `randomForest()` function:

```
library(MASS)
set.seed(1)
train <- sample(1:nrow(Boston), nrow(Boston)/2)
X.train <- Boston[train, -14]
X.test <- Boston[-train, -14]
Y.train <- Boston[train, 14]
Y.test <- Boston[-train, 14]
```

- a) Based on the plot, does the default setting of `ntree=500` seem reasonable to you?
- b) Based on the plot, would you prefer a random forest over a bagged ensemble?
- c) Does the OOB error give a more realistic estimate of test error for bagged ensembles or for random forests? Can you explain this?

Exercise 5: Fitting trees and ensembles for classification

Make exercise 8.8 a through e from ISLR.

Additional question:

Also fit a boosted tree ensemble. Compare the test MSE for the original tree, pruned tree, bagging, random forest and boosting.