Tree models

Rahul Unnikrishnan Nair

```
EnsurePackage<-function(x)
{ # EnsurePackage(x) - Installs and loads a package
  # if necessary
  x <- as.character(x)
  if (!require(x, character.only=TRUE))
  {
    install.packages(pkgs=x,
                     repos="http://cran.r-project.org")
  }
  require(x, character.only=TRUE)

}

#Installs and loads all packages necessary
#
Prepare.models<-function(){

  EnsurePackage("ISLR")
  EnsurePackage("MASS")
  EnsurePackage("mosaic")
  EnsurePackage("mosaicData")
  EnsurePackage("rpart")
  EnsurePackage("tidyr")
}

Prepare.models()
```

```
## Loading required package: ISLR
## Loading required package: MASS
## Loading required package: mosaic
## Loading required package: car
## Loading required package: dplyr
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:MASS':
##
##     select
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
```

```
##
##     intersect, setdiff, setequal, union
##
## Loading required package: lattice
## Loading required package: ggplot2
##
## Attaching package: 'mosaic'
##
## The following objects are masked from 'package:dplyr':
##
##     count, do, tally
##
## The following object is masked from 'package:car':
##
##     logit
##
## The following objects are masked from 'package:stats':
##
##     binom.test, cor, cov, D, fivenum, IQR, median, prop.test,
##     quantile, sd, t.test, var
##
## The following objects are masked from 'package:base':
##
##     max, mean, min, prod, range, sample, sum
##
## Loading required package: mosaicData
## Loading required package: rpart
## Loading required package: tidyr
```

```
library(ElemStatLearn)
data(spam)
# Exploratory Analysis
?spam
ncol(spam)
```

```
## [1] 58
```

```
head(spam)
```

```
##    A.1  A.2  A.3 A.4  A.5  A.6  A.7  A.8  A.9 A.10 A.11 A.12 A.13 A.14
## 1 0.00 0.64 0.64   0 0.32 0.00 0.00 0.00 0.00 0.00 0.00 0.64 0.00 0.00
## 2 0.21 0.28 0.50   0 0.14 0.28 0.21 0.07 0.00 0.94 0.21 0.79 0.65 0.21
## 3 0.06 0.00 0.71   0 1.23 0.19 0.19 0.12 0.64 0.25 0.38 0.45 0.12 0.00
## 4 0.00 0.00 0.00   0 0.63 0.00 0.31 0.63 0.31 0.63 0.31 0.31 0.31 0.00
## 5 0.00 0.00 0.00   0 0.63 0.00 0.31 0.63 0.31 0.63 0.31 0.31 0.31 0.00
## 6 0.00 0.00 0.00   0 1.85 0.00 0.00 1.85 0.00 0.00 0.00 0.00 0.00 0.00
##   A.15 A.16 A.17 A.18 A.19 A.20 A.21 A.22 A.23 A.24 A.25 A.26 A.27 A.28
## 1 0.00 0.32 0.00 1.29 1.93 0.00 0.96    0 0.00 0.00    0    0    0    0
## 2 0.14 0.14 0.07 0.28 3.47 0.00 1.59    0 0.43 0.43    0    0    0    0
## 3 1.75 0.06 0.06 1.03 1.36 0.32 0.51    0 1.16 0.06    0    0    0    0
## 4 0.00 0.31 0.00 0.00 3.18 0.00 0.31    0 0.00 0.00    0    0    0    0
```

```
## 5 0.00 0.31 0.00 0.00 3.18 0.00 0.31    0 0.00 0.00    0    0    0    0
## 6 0.00 0.00 0.00 0.00 0.00 0.00 0.00    0 0.00 0.00    0    0    0    0
##   A.29 A.30 A.31 A.32 A.33 A.34 A.35 A.36 A.37 A.38 A.39 A.40 A.41 A.42
## 1    0    0    0    0    0    0    0    0 0.00    0    0 0.00    0    0
## 2    0    0    0    0    0    0    0    0 0.07    0    0 0.00    0    0
## 3    0    0    0    0    0    0    0    0 0.00    0    0 0.06    0    0
## 4    0    0    0    0    0    0    0    0 0.00    0    0 0.00    0    0
## 5    0    0    0    0    0    0    0    0 0.00    0    0 0.00    0    0
## 6    0    0    0    0    0    0    0    0 0.00    0    0 0.00    0    0
##   A.43 A.44 A.45 A.46 A.47 A.48 A.49  A.50 A.51  A.52  A.53  A.54  A.55
## 1 0.00    0 0.00 0.00    0    0 0.00 0.000    0 0.778 0.000 0.000 3.756
## 2 0.00    0 0.00 0.00    0    0 0.00 0.132    0 0.372 0.180 0.048 5.114
## 3 0.12    0 0.06 0.06    0    0 0.01 0.143    0 0.276 0.184 0.010 9.821
## 4 0.00    0 0.00 0.00    0    0 0.00 0.137    0 0.137 0.000 0.000 3.537
## 5 0.00    0 0.00 0.00    0    0 0.00 0.135    0 0.135 0.000 0.000 3.537
## 6 0.00    0 0.00 0.00    0    0 0.00 0.223    0 0.000 0.000 0.000 3.000
##   A.56 A.57 spam
## 1   61  278 spam
## 2  101 1028 spam
## 3  485 2259 spam
## 4   40  191 spam
## 5   40  191 spam
## 6   15   54 spam
```

```
table(spam[58])
```

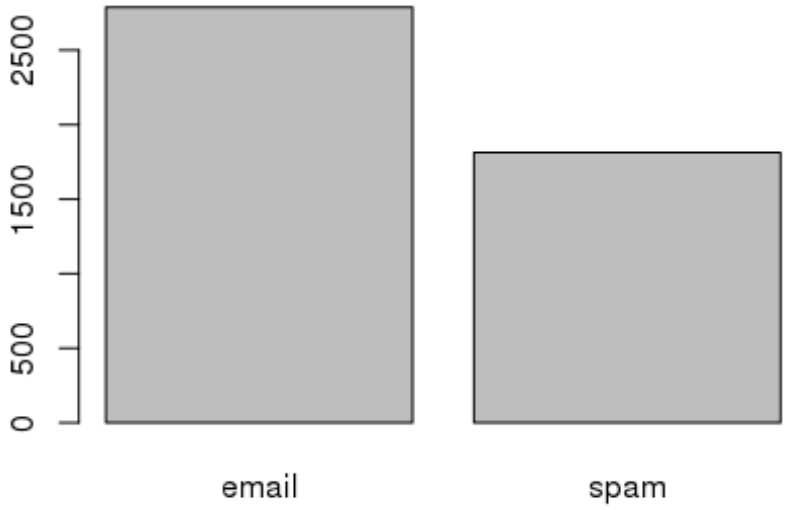
```
##
## email  spam
##  2788  1813
```

```
summary(spam[58]
      )
```

```
##     spam
##  email:2788
##  spam :1813
```

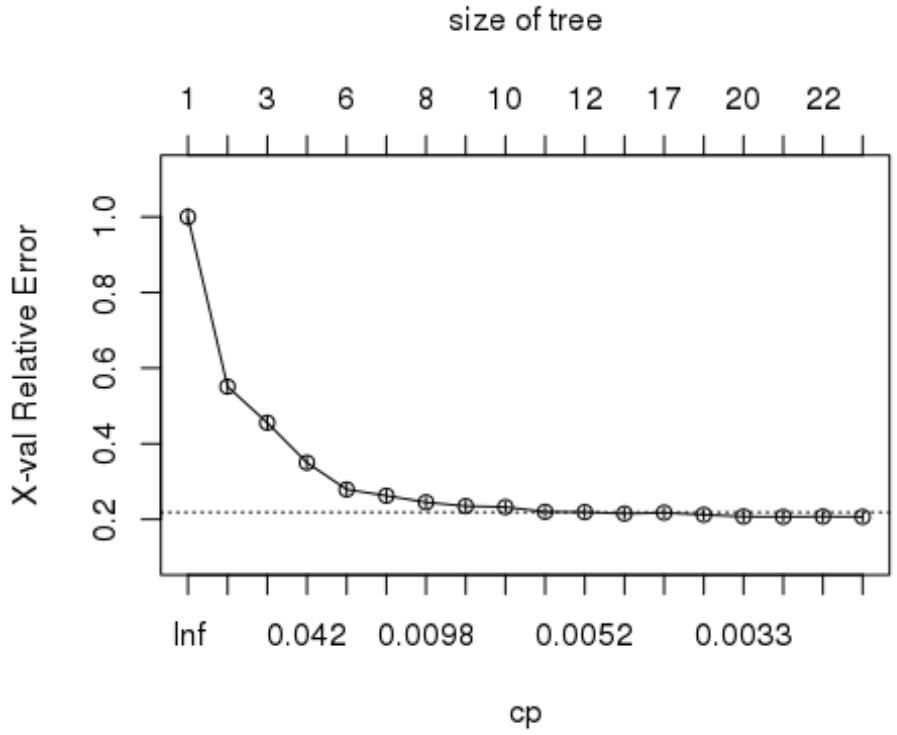
```
# description of the dataset

# This is a quantitative collection of 4601 observations of emails which contain spam emails and non spam emails. A total of 2788 non spam emails and 1813 spam emails. Each row is description of an email, with precentages of key words in the email as attributes among other things. The total number of attributes of 'spam' dataset is 58 among this the first 57 are numerical vectors while the last column contains a factor label, that describes if the mail is a spam or not.

#The first 48 columns contain continuous real attributes,[0,100], which describes the percentage of words in the email that match a WORD in the email. What the word is can be given as the name of the attribute. A standard format is to use word\_freq\_WORD. The next 6 attributes are also
```

```
real and continous,[0,100], that gives percentage of number of characters in the email that match a particular CHAR. The next 3 attributes are continuous real variables, [1,...], which describe average length of uninterupted sequence of capital letters, length of the longest uninterrupted sequence of capital letters, total number of capital letters in the e-mail.

#There are no null values in this dataset.

# plot of spam/non-spam
count.table <- as.data.frame(table(spam[58]))
?plot

plot(spam$spam)
```



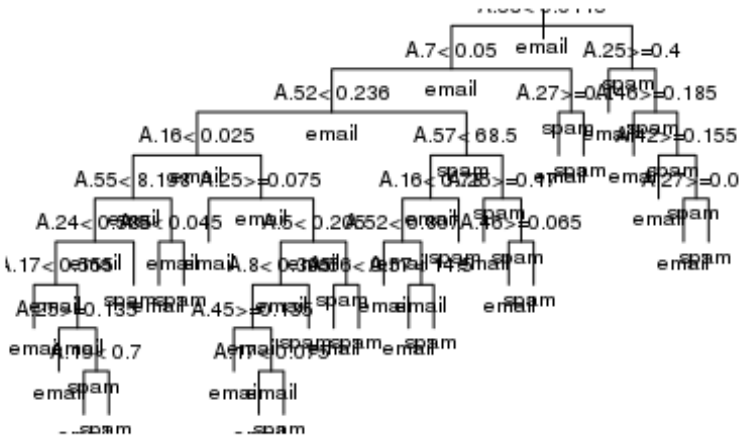

```
# splitting data into training and testing

set.seed(123)
?sample

## Help on topic 'sample' was found in the following packages:
##
##   Package             Library
##   base                /usr/lib64/R/library
##   dplyr               /usr/lib64/R/library
##   mosaic              /usr/lib64/R/library
##
```

```
##
## Using the first match ...

ind.spam <- sample(2, nrow(spam), replace = TRUE, prob = c(0.7, 0.3))
spam.train <- spam[ind.spam ==1,]
spam.test <- spam[ind.spam ==2,]


# creating a model using rpart
spam.model.rpart <- rpart(spam ~ ., method = "class", data = spam.train,
cp = 0.001)

plotcp(spam.model.rpart)
```




```
printcp(spam.model.rpart)

##
## Classification tree:
## rpart(formula = spam ~ ., data = spam.train, method = "class",
##     cp = 0.001)
##
## Variables actually used in tree construction:
##  [1] A.16 A.17 A.19 A.24 A.25 A.27 A.42 A.45 A.46 A.5  A.52 A.53 A.55
A.56
## [15] A.57 A.7  A.8
##
## Root node error: 1269/3229 = 0.393
```

```
##
## n= 3229
##
##          CP nsplit rel error  xerror     xstd
## 1  0.4751773      0   1.00000 1.00000 0.021871
## 2  0.1536643      1   0.52482 0.55083 0.018442
## 3  0.0425532      2   0.37116 0.45548 0.017166
## 4  0.0409771      3   0.32861 0.34988 0.015421
## 5  0.0141844      5   0.24665 0.27896 0.013990
## 6  0.0102443      6   0.23247 0.26241 0.013618
## 7  0.0094563      7   0.22222 0.24507 0.013211
## 8  0.0078802      8   0.21277 0.23483 0.012960
## 9  0.0063042      9   0.20489 0.23247 0.012902
## 10 0.0055162     10   0.19858 0.21986 0.012581
## 11 0.0049908     11   0.19307 0.21907 0.012561
## 12 0.0047281     14   0.17809 0.21513 0.012458
## 13 0.0039401     16   0.16864 0.21749 0.012520
## 14 0.0035461     17   0.16470 0.21198 0.012374
## 15 0.0031521     19   0.15760 0.20725 0.012248
## 16 0.0023641     20   0.15445 0.20646 0.012227
## 17 0.0015760     21   0.15209 0.20725 0.012248
## 18 0.0010000     27   0.14263 0.20646 0.012227

# minimum xerror from cptable = 0.21671, correponding cp = 0.0015760

# pruning the tree with the correponding cp
spam.model.pruned <- prune(spam.model.rpart, cp = 0.00157)
plot(spam.model.pruned, uniform = TRUE, main = "Classification Tree Model
using rpart")
text(spam.model.pruned, all = TRUE, cex = 0.65)
```




```
print(spam.model.pruned)
```

```
## n= 3229
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 3229 1269 email (0.60699907 0.39300093)
##     2) A.53< 0.0445 2398  552 email (0.76980817 0.23019183)
##       4) A.7< 0.05 2173  342 email (0.84261390 0.15738610)
##         8) A.52< 0.236 1794  151 email (0.91583055 0.08416945)
##          16) A.16< 0.025 1605   93 email (0.94205607 0.05794393)
##            32) A.55< 8.198 1581   81 email (0.94876660 0.05123340)
##              64) A.24< 0.585 1566   73 email (0.95338442 0.04661558)
##               128) A.17< 0.555 1516   61 email (0.95976253 0.04023747) *
##               129) A.17>=0.555 50   12 email (0.76000000 0.24000000)
##                 258) A.25>=0.135 28    0 email (1.00000000 0.00000000) *
##                 259) A.25< 0.135 22   10 spam (0.45454545 0.54545455)
##                   518) A.19< 0.7 7    1 email (0.85714286 0.14285714) *
##                   519) A.19>=0.7 15    4 spam (0.26666667 0.73333333) *
##              65) A.24>=0.585 15    7 spam (0.46666667 0.53333333) *
##            33) A.55>=8.198 24   12 email (0.50000000 0.50000000)
##              66) A.5< 0.045 15    3 email (0.80000000 0.20000000) *
##              67) A.5>=0.045 9    0 spam (0.00000000 1.00000000) *
##          17) A.16>=0.025 189   58 email (0.69312169 0.30687831)
##            34) A.25>=0.075 53    1 email (0.98113208 0.01886792) *
##            35) A.25< 0.075 136   57 email (0.58088235 0.41911765)
```

```
##           70) A.5< 0.205 91   25 email (0.72527473 0.27472527)
##           140) A.8< 0.395 84   19 email (0.77380952 0.22619048)
##             280) A.45>=0.135 30    0 email (1.00000000 0.00000000) *
##             281) A.45< 0.135 54   19 email (0.64814815 0.35185185)
##               562) A.17< 0.075 42   11 email (0.73809524 0.26190476) *
##               563) A.17>=0.075 12    4 spam (0.33333333 0.66666667) *
##           141) A.8>=0.395 7    1 spam (0.14285714 0.85714286) *
##          71) A.5>=0.205 45   13 spam (0.28888889 0.71111111)
##           142) A.56< 9 14    5 email (0.64285714 0.35714286) *
##           143) A.56>=9 31    4 spam (0.12903226 0.87096774) *
##       9) A.52>=0.236 379  188 spam (0.49604222 0.50395778)
##        18) A.57< 68.5 197   48 email (0.75634518 0.24365482)
##          36) A.16< 0.78 179   33 email (0.81564246 0.18435754)
##            72) A.52< 0.807 130   10 email (0.92307692 0.07692308) *
##            73) A.52>=0.807 49   23 email (0.53061224 0.46938776)
##            146) A.57< 14.5 23    4 email (0.82608696 0.17391304) *
##            147) A.57>=14.5 26    7 spam (0.26923077 0.73076923) *
##          37) A.16>=0.78 18    3 spam (0.16666667 0.83333333) *
##        19) A.57>=68.5 182   39 spam (0.21428571 0.78571429)
##          38) A.25>=0.17 20    1 email (0.95000000 0.05000000) *
##          39) A.25< 0.17 162   20 spam (0.12345679 0.87654321)
##            78) A.46>=0.065 8    0 email (1.00000000 0.00000000) *
##            79) A.46< 0.065 154   12 spam (0.07792208 0.92207792) *
##      5) A.7>=0.05 225   15 spam (0.06666667 0.93333333)
##       10) A.27>=0.14 7    0 email (1.00000000 0.00000000) *
##       11) A.27< 0.14 218    8 spam (0.03669725 0.96330275) *
##     3) A.53>=0.0445 831  114 spam (0.13718412 0.86281588)
##       6) A.25>=0.4 62    4 email (0.93548387 0.06451613) *
##       7) A.25< 0.4 769   56 spam (0.07282185 0.92717815)
##        14) A.46>=0.185 21    4 email (0.80952381 0.19047619) *
##        15) A.46< 0.185 748   39 spam (0.05213904 0.94786096)
##          30) A.42>=0.155 12    1 email (0.91666667 0.08333333) *
##          31) A.42< 0.155 736   28 spam (0.03804348 0.96195652)
##            62) A.27>=0.08 7    2 email (0.71428571 0.28571429) *
##            63) A.27< 0.08 729   23 spam (0.03155007 0.96844993) *
```

```
# prediction
spam.predict.rpart <-predict(spam.model.pruned, newdata=spam.test, type
= 'class')
table(spam.predict.rpart, spam.test$spam)
```

```
##
## spam.predict.rpart email spam
##            email   772   73
##            spam     56  471
```

```
mean(spam.predict.rpart == spam.test$spam) # classification mean
```

```
## [1] 0.9059767
```

```
rpart.error <- mean(spam.predict.rpart != spam.test$spam) # miss
classification mean - testing error -rpart
```

```
rpart.error
```

```
## [1] 0.09402332
```