patient8

Anuhya B S

2022-06-09

```
#Importing libraries
library('R.matlab')
```

```
## R.matlab v3.6.2 (2018-09-26) successfully loaded. See ?R.matlab for help.
```

```
##
## Attaching package: 'R.matlab'
```

```
## The following objects are masked from 'package:base':
##
##     getOption, isOpen
```

```
library(caTools)
library(e1071)
library(class)
library(tree)
library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
#Loading Data
p1 <- readMat("data-science-P8.mat")

info <- as.data.frame(p1[2])
info <- t(info)
info <- as.data.frame(info)
lab.grp <-as.data.frame(matrix(nrow=0,ncol=1))
lab.wrd <-as.data.frame(matrix(nrow=0,ncol=1))
for (i in 1:360){
  lab.grp <-  rbind(lab.grp,info$cond[[i]])
  lab.wrd <-  rbind(lab.wrd,info$word[[i]])
}

p1.data <- p1$data
voxels <-as.data.frame(matrix(nrow=0,ncol=21764))
for (i in 1:360){
  voxels <-  rbind(voxels,p1.data[[i]][[1]])
}
```

```
# Principal Component Analysis for Feature Reduction
pr.out <- prcomp(voxels)
cumsum((pr.out$sdev^2)/sum(pr.out$sdev^2))
```

```
##   [1] 0.1600838 0.2984576 0.4082840 0.4896976 0.5481297 0.5883898 0.6191093
```

```
##   [8] 0.6455275 0.6695337 0.6905560 0.7062618 0.7208647 0.7340099 0.7451037
##  [15] 0.7558362 0.7653256 0.7741827 0.7819767 0.7887445 0.7950136 0.8012260
##  [22] 0.8069486 0.8124292 0.8173665 0.8222744 0.8267839 0.8310863 0.8350850
##  [29] 0.8387262 0.8422690 0.8457084 0.8490501 0.8521518 0.8551919 0.8581085
##  [36] 0.8608033 0.8633853 0.8659205 0.8682628 0.8705257 0.8727467 0.8748607
##  [43] 0.8768983 0.8788686 0.8807391 0.8825528 0.8842897 0.8859211 0.8875287
##  [50] 0.8890559 0.8905138 0.8919343 0.8933210 0.8946999 0.8960519 0.8973162
##  [57] 0.8985505 0.8997674 0.9009606 0.9021495 0.9032775 0.9043907 0.9054806
##  [64] 0.9065384 0.9075832 0.9086221 0.9096348 0.9106292 0.9115899 0.9125382
##  [71] 0.9134620 0.9143648 0.9152642 0.9161453 0.9169990 0.9178420 0.9186536
##  [78] 0.9194578 0.9202554 0.9210485 0.9218289 0.9225846 0.9233284 0.9240605
##  [85] 0.9247773 0.9254828 0.9261796 0.9268702 0.9275526 0.9282180 0.9288791
##  [92] 0.9295293 0.9301700 0.9308005 0.9314235 0.9320330 0.9326348 0.9332218
##  [99] 0.9337982 0.9343682 0.9349323 0.9354906 0.9360419 0.9365842 0.9371225
## [106] 0.9376586 0.9381834 0.9387060 0.9392198 0.9397260 0.9402270 0.9407203
## [113] 0.9412070 0.9416885 0.9421641 0.9426327 0.9430962 0.9435585 0.9440145
## [120] 0.9444677 0.9449161 0.9453588 0.9457986 0.9462345 0.9466688 0.9471006
## [127] 0.9475237 0.9479407 0.9483552 0.9487670 0.9491760 0.9495787 0.9499805
## [134] 0.9503752 0.9507691 0.9511581 0.9515461 0.9519321 0.9523132 0.9526894
## [141] 0.9530602 0.9534302 0.9537985 0.9541623 0.9545227 0.9548815 0.9552373
## [148] 0.9555919 0.9559428 0.9562929 0.9566387 0.9569822 0.9573248 0.9576636
## [155] 0.9580011 0.9583345 0.9586655 0.9589952 0.9593225 0.9596474 0.9599687
## [162] 0.9602895 0.9606087 0.9609249 0.9612381 0.9615502 0.9618610 0.9621704
## [169] 0.9624783 0.9627824 0.9630851 0.9633876 0.9636879 0.9639862 0.9642840
## [176] 0.9645794 0.9648739 0.9651676 0.9654584 0.9657483 0.9660348 0.9663187
## [183] 0.9666020 0.9668838 0.9671637 0.9674418 0.9677187 0.9679942 0.9682675
## [190] 0.9685404 0.9688114 0.9690811 0.9693495 0.9696143 0.9698787 0.9701421
## [197] 0.9704046 0.9706647 0.9709244 0.9711816 0.9714373 0.9716921 0.9719461
## [204] 0.9721980 0.9724483 0.9726980 0.9729454 0.9731925 0.9734376 0.9736819
## [211] 0.9739254 0.9741682 0.9744089 0.9746489 0.9748885 0.9751263 0.9753632
## [218] 0.9755996 0.9758339 0.9760676 0.9762993 0.9765306 0.9767611 0.9769906
## [225] 0.9772185 0.9774460 0.9776727 0.9778975 0.9781212 0.9783442 0.9785665
## [232] 0.9787875 0.9790077 0.9792265 0.9794446 0.9796620 0.9798783 0.9800937
## [239] 0.9803092 0.9805240 0.9807369 0.9809491 0.9811611 0.9813718 0.9815813
## [246] 0.9817899 0.9819978 0.9822054 0.9824112 0.9826154 0.9828193 0.9830217
## [253] 0.9832239 0.9834257 0.9836263 0.9838263 0.9840254 0.9842243 0.9844221
## [260] 0.9846188 0.9848150 0.9850099 0.9852038 0.9853976 0.9855905 0.9857828
## [267] 0.9859733 0.9861635 0.9863533 0.9865420 0.9867304 0.9869170 0.9871028
## [274] 0.9872878 0.9874715 0.9876545 0.9878366 0.9880181 0.9881990 0.9883794
## [281] 0.9885592 0.9887386 0.9889173 0.9890956 0.9892723 0.9894485 0.9896240
## [288] 0.9897991 0.9899737 0.9901478 0.9903209 0.9904935 0.9906653 0.9908364
## [295] 0.9910068 0.9911763 0.9913454 0.9915136 0.9916806 0.9918465 0.9920119
## [302] 0.9921768 0.9923413 0.9925050 0.9926677 0.9928293 0.9929901 0.9931504
## [309] 0.9933102 0.9934690 0.9936270 0.9937844 0.9939407 0.9940965 0.9942517
## [316] 0.9944063 0.9945592 0.9947115 0.9948638 0.9950151 0.9951650 0.9953138
## [323] 0.9954617 0.9956092 0.9957562 0.9959017 0.9960461 0.9961895 0.9963325
## [330] 0.9964743 0.9966142 0.9967536 0.9968928 0.9970305 0.9971666 0.9973015
## [337] 0.9974343 0.9975666 0.9976963 0.9978253 0.9979521 0.9980770 0.9982010
## [344] 0.9983237 0.9984457 0.9985657 0.9986843 0.9988019 0.9989174 0.9990328
## [351] 0.9991463 0.9992586 0.9993697 0.9994791 0.9995874 0.9996937 0.9997975
## [358] 0.9999001 1.0000000 1.0000000
```

```
pcs <- as.data.frame(pr.out$x[,1:300])
pcs$grp <- lab.grp$V1
```

```
#pcs$wrd <- lab.wrd$V1
```

```
# Splitting data into training and test data
set.seed(100)
#sample <- sample(1:nrow(pcs), 300)
pcs.train <- pcs[1:300,]
pcs.test <- pcs[301:360,]
#pcs.train <- subset(pcs, sample == TRUE)
#pcs.test  <- subset(pcs, sample == FALSE)
pcs.train.x <- subset(pcs.train, select = -c(grp))
pcs.train.labs <- pcs.train$grp
pcs.test.x <- subset(pcs.test, select = -c(grp))
pcs.test.labs <- pcs.test$grp
```

```
# Classification Algorithms

# Naive Bayes Classifier

nb.fit <- naiveBayes(grp ~ . , data = pcs.train)
nb.class <- predict(nb.fit,pcs.test.x)
nb.class
```

```
##  [1] tool      bodypart  tool      vehicle   bodypart  animal    animal
##  [8] animal    insect    tool      insect    vehicle   animal    animal
## [15] animal    vehicle   insect    vehicle   animal    kitchen   furniture
## [22] tool      vehicle   kitchen   vehicle   animal    animal    manmade
## [29] furniture animal    vehicle   animal    vehicle   animal    vehicle
## [36] vehicle   animal    kitchen   vehicle   kitchen   animal    bodypart
## [43] furniture animal    animal    animal    vehicle   vehicle   animal
## [50] vehicle   vehicle   vehicle   vehicle   animal    furniture vehicle
## [57] furniture vehicle   vehicle   vehicle
## 12 Levels: animal bodypart building buildpart clothing furniture ... vehicle
```

```
confusion_mat.nb = as.matrix(table(Actual_Values = pcs.test.labs, Predicted_Values = nb.class))
print(confusion_mat.nb)
```

```
##              Predicted_Values
## Actual_Values animal bodypart building buildpart clothing furniture insect
##     animal         0        0        0         0        0         0      1
##     bodypart       2        1        0         0        0         0      0
##     building       4        0        0         0        0         1      0
##     buildpart      1        0        0         0        0         1      0
##     clothing       1        1        0         0        0         0      0
##     furniture      2        0        0         0        0         0      1
##     insect         2        0        0         0        0         1      1
##     kitchen        3        1        0         0        0         0      0
##     manmade        1        0        0         0        0         1      0
##     tool           1        0        0         0        0         0      0
##     vegetable      2        0        0         0        0         0      0
##     vehicle        0        0        0         0        0         1      0
##              Predicted_Values
## Actual_Values kitchen manmade tool vegetable vehicle
##     animal          0       1    1         0       2
##     bodypart        0       0    0         0       2
##     building        0       0    0         0       0
```

```
##     buildpart       0       0    0         0       3
##     clothing        0       0    1         0       2
##     furniture       0       0    0         0       2
##     insect          0       0    1         0       0
##     kitchen         0       0    0         0       1
##     manmade         0       0    1         0       2
##     tool            2       0    0         0       2
##     vegetable       1       0    0         0       2
##     vehicle         1       0    0         0       3
```

```
print(mean(nb.class == pcs.test$grp))
```

```
## [1] 0.08333333
```

```
# KNN

knn.pred <- knn(pcs.train.x, pcs.test.x, pcs.train.labs, k=5)

confusion_mat.knn = as.matrix(table(pcs.test.labs, knn.pred))
print(confusion_mat.knn)
```

```
##              knn.pred
## pcs.test.labs animal bodypart building buildpart clothing furniture insect
##     animal         0        1        1         2        0         0      0
##     bodypart       0        0        0         0        1         0      0
##     building       1        0        1         1        0         1      0
##     buildpart      0        0        0         1        1         0      1
##     clothing       0        0        0         0        1         1      0
##     furniture      1        0        1         0        0         0      0
##     insect         1        0        0         1        1         0      1
##     kitchen        0        1        0         0        0         0      0
##     manmade        2        0        1         0        0         0      0
##     tool           0        1        0         2        0         0      1
##     vegetable      0        0        0         0        1         1      1
##     vehicle        1        1        1         1        1         0      0
##              knn.pred
## pcs.test.labs kitchen manmade tool vegetable vehicle
##     animal          0       0    1         0       0
##     bodypart        1       0    1         1       1
##     building        0       0    0         1       0
##     buildpart       0       1    0         0       1
##     clothing        1       0    1         0       1
##     furniture       1       0    1         0       1
##     insect          0       0    0         0       1
##     kitchen         1       1    1         0       1
##     manmade         0       0    1         1       0
##     tool            1       0    0         0       0
##     vegetable       0       0    0         1       1
##     vehicle         0       0    0         0       0
```

```
print(mean(knn.pred == pcs.test$grp))
```

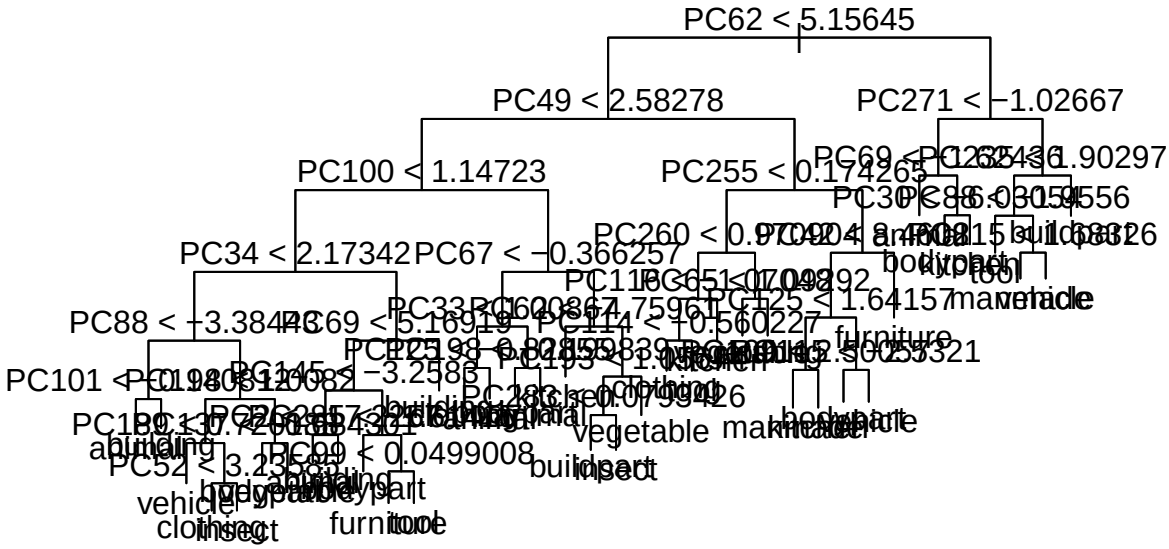
```
## [1] 0.1
```

```
# Decision Trees
set.seed(100)
tree.fit <- tree(as.factor(grp) ~ ., data = pcs.train)
```

```
summary(tree.fit)
```

```
##
## Classification tree:
## tree(formula = as.factor(grp) ~ ., data = pcs.train)
## Variables actually used in tree construction:
##  [1] "PC62"  "PC49"  "PC100" "PC34"  "PC88"  "PC101" "PC198" "PC189" "PC52"
## [10] "PC137" "PC69"  "PC145" "PC2"   "PC285" "PC99"  "PC67"  "PC33"  "PC125"
## [19] "PC193" "PC283" "PC255" "PC260" "PC116" "PC114" "PC65"  "PC42"  "PC109"
## [28] "PC115" "PC271" "PC30"  "PC235" "PC215"
## Number of terminal nodes:  38
## Residual mean deviance:  1.971 = 516.4 / 262
## Misclassification error rate: 0.38 = 114 / 300
```

```
plot(tree.fit)
text(tree.fit, pretty = 0)
```




```
tree.pred <- predict(tree.fit, newdata = pcs.test, type = "class")
tree.pred
```

```
##  [1] animal    vegetable animal    building  animal    tool      kitchen
##  [8] kitchen   clothing  kitchen   vehicle   vehicle   vegetable animal
## [15] kitchen   furniture vehicle   tool      manmade   building  buildpart
## [22] clothing  tool      tool      manmade   kitchen   clothing  furniture
## [29] insect    kitchen   building  buildpart kitchen   clothing  building
## [36] clothing  vegetable bodypart  tool      furniture kitchen   manmade
## [43] vehicle   bodypart  kitchen   tool      vehicle   animal    tool
```

```
## [50] furniture kitchen   kitchen   tool      manmade   furniture furniture
## [57] vehicle   vegetable vehicle   furniture
## 12 Levels: animal bodypart building buildpart clothing furniture ... vehicle
```

```
confusion_mat.dt = as.matrix(table(pcs.test.labs, tree.pred))
print(confusion_mat.dt)
```

```
##              tree.pred
## pcs.test.labs animal bodypart building buildpart clothing furniture insect
##     animal         1        0        0         0        0         1      0
##     bodypart       1        1        0         0        0         1      0
##     building       1        0        0         0        0         0      0
##     buildpart      0        0        1         0        0         2      0
##     clothing       1        0        1         0        1         0      0
##     furniture      0        0        1         0        1         0      0
##     insect         1        0        0         1        2         0      0
##     kitchen        0        0        0         1        0         1      0
##     manmade        0        0        0         0        1         0      0
##     tool           0        0        0         0        0         1      0
##     vegetable      0        1        0         0        0         0      0
##     vehicle        0        0        1         0        0         1      1
##              tree.pred
## pcs.test.labs kitchen manmade tool vegetable vehicle
##     animal          0       0    0         1       2
##     bodypart        0       1    1         0       0
##     building        1       0    1         1       1
##     buildpart       0       1    1         0       0
##     clothing        2       0    0         0       0
##     furniture       1       0    0         0       2
##     insect          0       0    0         1       0
##     kitchen         1       0    1         1       0
##     manmade         1       1    1         0       1
##     tool            2       0    1         0       1
##     vegetable       3       1    0         0       0
##     vehicle         0       0    2         0       0
```

```
print(mean(tree.pred == pcs.test$grp))
```

```
## [1] 0.1
```

```
# Random Forest

rf.fit <- randomForest(as.factor(grp) ~ ., data = pcs.train,, mtry = 80, importance = TRUE)
summary(rf.fit)
```

```
##                Length Class  Mode
## call              6   -none- call
## type              1   -none- character
## predicted       300   factor numeric
## err.rate       6500   -none- numeric
## confusion       156   -none- numeric
## votes          3600   matrix numeric
## oob.times       300   -none- numeric
## classes          12   -none- character
## importance     4200   -none- numeric
## importanceSD   3900   -none- numeric
```

```
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            14   -none- list
## y                300   factor numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
## terms              3   terms  call
```

```
rf.pred <- predict(rf.fit, newdata = pcs.test, type = "class")
confusion_mat.rf = as.matrix(table(pcs.test.labs, rf.pred))
print(confusion_mat.rf)
```

```
##              rf.pred
## pcs.test.labs animal bodypart building buildpart clothing furniture insect
##     animal         0        0        0         0        1         2      1
##     bodypart       0        2        2         0        0         0      0
##     building       0        0        0         0        1         1      0
##     buildpart      0        0        0         0        0         0      0
##     clothing       0        1        1         0        1         0      0
##     furniture      0        1        2         0        0         0      0
##     insect         0        0        1         0        2         0      0
##     kitchen        1        0        1         0        0         0      0
##     manmade        0        0        0         0        0         2      0
##     tool           0        0        0         0        1         1      0
##     vegetable      0        0        1         0        1         0      1
##     vehicle        1        1        0         0        0         0      0
##              rf.pred
## pcs.test.labs kitchen manmade tool vegetable vehicle
##     animal          0       0    0         1       0
##     bodypart        0       0    0         0       1
##     building        1       1    0         0       1
##     buildpart       2       1    0         0       2
##     clothing        1       0    1         0       0
##     furniture       1       0    0         1       0
##     insect          0       0    0         0       2
##     kitchen         1       1    1         0       0
##     manmade         1       1    0         1       0
##     tool            1       0    0         0       2
##     vegetable       0       1    0         0       1
##     vehicle         1       1    0         0       1
```

```
print(mean(rf.pred == pcs.test$grp))
```

```
## [1] 0.1
```

```
manmade <- c("furniture", "clothing", "manmade", "tool", "kitchen", "vehicle", "building", "buildpart")
natural <- c("insect", "animal", "vegetable", "bodypart")
df_new <- within(pcs, {
 cls <- "manmade"
 cls[grp %in% manmade] <- "manmade"
 cls[grp %in% natural] <- "natural"
})
pcs$cls <- df_new$cls
```

```r
# Splitting data into training and test data
set.seed(100)
#sample <- sample(1:nrow(pcs), 300)
pcs.train <- pcs[1:300,]
pcs.test <- pcs[301:360,]
#pcs.train <- subset(pcs, sample == TRUE)
#pcs.test  <- subset(pcs, sample == FALSE)
pcs.train.x <- subset(pcs.train, select = -c(grp,cls))
pcs.train.labs <- pcs.train$cls
pcs.test.x <- subset(pcs.test, select = -c(grp,cls))
pcs.test.labs <- pcs.test$cls
```

```r
# Classification Algorithms

# Naive Bayes Classifier

nb.fit <- naiveBayes(cls ~ . , data = pcs.train)
nb.class <- predict(nb.fit,pcs.test.x)
nb.class
```

```
##  [1] manmade natural manmade manmade manmade manmade natural natural natural
## [10] manmade manmade manmade manmade natural natural manmade manmade natural
## [19] manmade manmade natural natural natural natural manmade manmade manmade
## [28] manmade manmade natural manmade manmade manmade manmade natural manmade
## [37] natural natural manmade natural natural manmade manmade manmade natural
## [46] manmade manmade manmade natural natural natural manmade manmade manmade
## [55] natural manmade manmade manmade manmade manmade
## Levels: manmade natural
```

```r
confusion_mat.nb = as.matrix(table(Actual_Values = pcs.test.labs, Predicted_Values = nb.class))
print(confusion_mat.nb)
```

```
##              Predicted_Values
## Actual_Values manmade natural
##       manmade      24      16
##       natural      14       6
```

```r
print(mean(nb.class == pcs.test$cls))
```

```
## [1] 0.5
```

```r
# KNN

knn.pred <- knn(pcs.train.x, pcs.test.x, pcs.train.labs, k=3)

confusion_mat.knn = as.matrix(table(pcs.test.labs, knn.pred))
print(confusion_mat.knn)
```

```
##              knn.pred
## pcs.test.labs manmade natural
##       manmade      30      10
##       natural      17       3
```

```r
print(mean(knn.pred== pcs.test$cls))
```

```
## [1] 0.55
```

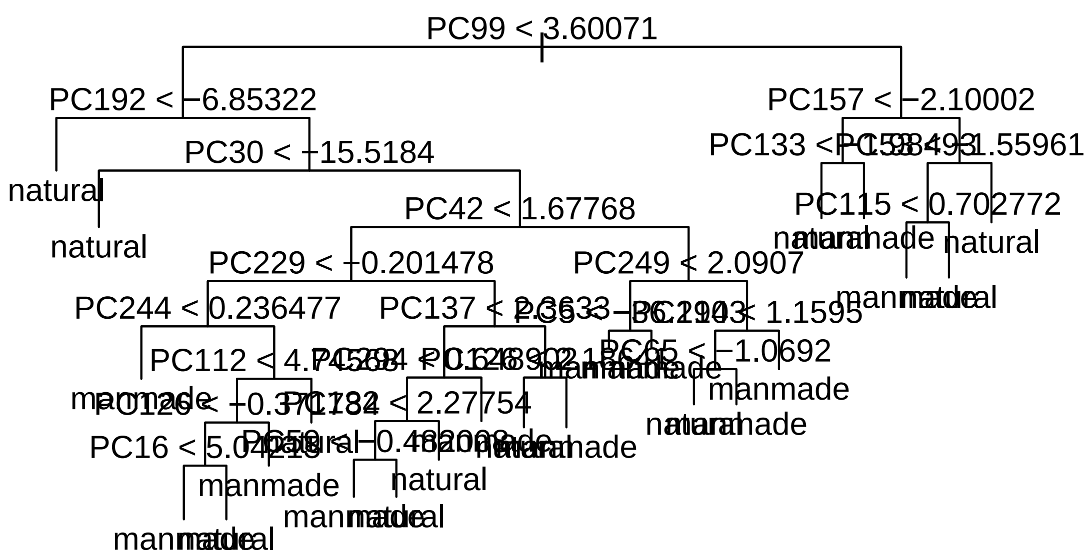
```r
# Decision Trees
set.seed(100)
tree.fit <- tree(as.factor(cls) ~ ., data = pcs.train)
```

```
## Warning in tree(as.factor(cls) ~ ., data = pcs.train): NAs introduced by
## coercion
```

```r
summary(tree.fit)
```

```
##
## Classification tree:
## tree(formula = as.factor(cls) ~ ., data = pcs.train)
## Variables actually used in tree construction:
##  [1] "PC99"  "PC192" "PC30"  "PC42"  "PC229" "PC244" "PC112" "PC126" "PC16"
## [10] "PC137" "PC294" "PC132" "PC59"  "PC249" "PC5"   "PC194" "PC65"  "PC157"
## [19] "PC133" "PC53"  "PC115"
## Number of terminal nodes:  23
## Residual mean deviance:  0.2055 = 56.92 / 277
## Misclassification error rate: 0.03333 = 10 / 300
```

```r
plot(tree.fit)
text(tree.fit, pretty = 0)
```




```r
tree.pred <- predict(tree.fit, newdata = pcs.test, type = "class")
```

```
## Warning in pred1.tree(object, tree.matrix(newdata)): NAs introduced by coercion
```

```
tree.pred
```

```
##  [1] manmade natural manmade manmade manmade manmade manmade natural manmade
## [10] manmade manmade manmade manmade natural natural natural natural manmade
## [19] manmade manmade natural natural manmade natural manmade manmade manmade
## [28] natural natural natural manmade manmade natural natural manmade manmade
## [37] manmade natural manmade manmade natural manmade natural natural manmade
## [46] manmade manmade manmade natural manmade natural manmade natural manmade
## [55] natural manmade manmade manmade manmade natural
## Levels: manmade natural
```

```
confusion_mat.dt = as.matrix(table(pcs.test.labs, tree.pred))
print(confusion_mat.dt)
```

```
##              tree.pred
## pcs.test.labs manmade natural
##       manmade      24      16
##       natural      13       7
```

```
print(mean(tree.pred== pcs.test$cls))
```

```
## [1] 0.5166667
```

```
# Random Forest

rf.fit <- randomForest(as.factor(cls) ~ ., data = pcs.train,, mtry = 80, importance = TRUE)
summary(rf.fit)
```

```
##                 Length Class  Mode
## call               6   -none- call
## type               1   -none- character
## predicted        300   factor numeric
## err.rate        1500   -none- numeric
## confusion          6   -none- numeric
## votes            600   matrix numeric
## oob.times        300   -none- numeric
## classes            2   -none- character
## importance      1204   -none- numeric
## importanceSD     903   -none- numeric
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            14   -none- list
## y                300   factor numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
## terms              3   terms  call
```

```
rf.pred <- predict(rf.fit, newdata = pcs.test, type = "class")
confusion_mat.rf = as.matrix(table(pcs.test.labs, rf.pred))
print(confusion_mat.rf)
```

```
##              rf.pred
## pcs.test.labs manmade natural
##       manmade      40       0
##       natural      10      10
```

```
print(mean(rf.pred== pcs.test$cls))
```

```
## [1] 0.8333333
```