patient7

Anuhya B S

2022-06-9

```
#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-P7.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] 0.3857170 0.4576886 0.5086031 0.5501438 0.5721940 0.5932694 0.6117646
```

```
##   [8] 0.6276242 0.6421856 0.6544695 0.6652107 0.6744389 0.6829009 0.6902380
##  [15] 0.6968845 0.7032281 0.7086449 0.7137225 0.7183948 0.7229486 0.7273105
##  [22] 0.7312377 0.7350798 0.7388233 0.7421700 0.7452920 0.7482914 0.7512064
##  [29] 0.7540148 0.7566585 0.7592562 0.7618062 0.7642516 0.7666668 0.7689422
##  [36] 0.7711221 0.7732860 0.7753782 0.7773633 0.7792801 0.7811221 0.7829534
##  [43] 0.7847505 0.7864916 0.7881979 0.7898646 0.7915231 0.7931189 0.7946916
##  [50] 0.7962388 0.7977647 0.7992667 0.8007372 0.8021777 0.8036064 0.8049892
##  [57] 0.8063417 0.8076857 0.8090149 0.8103283 0.8116255 0.8128862 0.8141257
##  [64] 0.8153565 0.8165815 0.8177741 0.8189408 0.8201039 0.8212558 0.8224016
##  [71] 0.8235300 0.8246407 0.8257440 0.8268351 0.8279180 0.8289961 0.8300577
##  [78] 0.8311130 0.8321618 0.8332024 0.8342331 0.8352583 0.8362774 0.8372922
##  [85] 0.8382914 0.8392820 0.8402649 0.8412356 0.8422049 0.8431661 0.8441244
##  [92] 0.8450710 0.8460131 0.8469430 0.8478698 0.8487833 0.8496930 0.8506009
##  [99] 0.8515002 0.8523946 0.8532862 0.8541744 0.8550573 0.8559364 0.8568108
## [106] 0.8576836 0.8585482 0.8594107 0.8602639 0.8611142 0.8619631 0.8628073
## [113] 0.8636462 0.8644814 0.8653080 0.8661328 0.8669564 0.8677717 0.8685858
## [120] 0.8693974 0.8702072 0.8710064 0.8718049 0.8726015 0.8733942 0.8741811
## [127] 0.8749604 0.8757371 0.8765116 0.8772827 0.8780506 0.8788179 0.8795821
## [134] 0.8803419 0.8810984 0.8818496 0.8826001 0.8833491 0.8840923 0.8848327
## [141] 0.8855692 0.8863029 0.8870360 0.8877652 0.8884926 0.8892173 0.8899394
## [148] 0.8906583 0.8913736 0.8920847 0.8927946 0.8935027 0.8942094 0.8949147
## [155] 0.8956143 0.8963132 0.8970118 0.8977070 0.8984010 0.8990934 0.8997845
## [162] 0.9004719 0.9011551 0.9018346 0.9025135 0.9031900 0.9038613 0.9045322
## [169] 0.9052016 0.9058696 0.9065326 0.9071936 0.9078543 0.9085105 0.9091659
## [176] 0.9098181 0.9104682 0.9111166 0.9117641 0.9124100 0.9130551 0.9136967
## [183] 0.9143377 0.9149756 0.9156112 0.9162463 0.9168774 0.9175078 0.9181370
## [190] 0.9187650 0.9193919 0.9200158 0.9206386 0.9212601 0.9218806 0.9224987
## [197] 0.9231149 0.9237295 0.9243418 0.9249518 0.9255589 0.9261647 0.9267696
## [204] 0.9273724 0.9279722 0.9285715 0.9291681 0.9297632 0.9303583 0.9309518
## [211] 0.9315441 0.9321339 0.9327218 0.9333085 0.9338932 0.9344762 0.9350566
## [218] 0.9356362 0.9362144 0.9367910 0.9373669 0.9379421 0.9385144 0.9390855
## [225] 0.9396539 0.9402211 0.9407865 0.9413498 0.9419120 0.9424729 0.9430314
## [232] 0.9435870 0.9441420 0.9446963 0.9452476 0.9457972 0.9463463 0.9468938
## [239] 0.9474405 0.9479847 0.9485284 0.9490684 0.9496077 0.9501439 0.9506793
## [246] 0.9512136 0.9517469 0.9522788 0.9528081 0.9533369 0.9538636 0.9543887
## [253] 0.9549124 0.9554349 0.9559559 0.9564747 0.9569932 0.9575096 0.9580250
## [260] 0.9585379 0.9590499 0.9595608 0.9600706 0.9605792 0.9610857 0.9615910
## [267] 0.9620947 0.9625973 0.9630973 0.9635965 0.9640947 0.9645908 0.9650863
## [274] 0.9655795 0.9660718 0.9665602 0.9670472 0.9675323 0.9680166 0.9685002
## [281] 0.9689821 0.9694639 0.9699416 0.9704189 0.9708934 0.9713669 0.9718400
## [288] 0.9723100 0.9727794 0.9732471 0.9737114 0.9741752 0.9746377 0.9750994
## [295] 0.9755584 0.9760170 0.9764739 0.9769302 0.9773836 0.9778366 0.9782871
## [302] 0.9787358 0.9791820 0.9796280 0.9800715 0.9805138 0.9809539 0.9813925
## [309] 0.9818298 0.9822653 0.9826988 0.9831304 0.9835610 0.9839904 0.9844160
## [316] 0.9848395 0.9852615 0.9856816 0.9860997 0.9865167 0.9869308 0.9873420
## [323] 0.9877520 0.9881591 0.9885633 0.9889651 0.9893560 0.9897447 0.9901256
## [330] 0.9905057 0.9908798 0.9912469 0.9916137 0.9919752 0.9923319 0.9926865
## [337] 0.9930394 0.9933843 0.9937286 0.9940702 0.9944059 0.9947398 0.9950709
## [344] 0.9953985 0.9957250 0.9960513 0.9963730 0.9966912 0.9970056 0.9973190
## [351] 0.9976276 0.9979340 0.9982400 0.9985409 0.9988396 0.9991367 0.9994291
## [358] 0.9997189 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] manmade   clothing  manmade   manmade   building  building  vegetable
##  [8] manmade   bodypart  vehicle   vehicle   vegetable manmade   vegetable
## [15] clothing  furniture vegetable bodypart  vegetable tool      animal
## [22] buildpart bodypart  buildpart vehicle   vehicle   clothing  animal
## [29] buildpart vehicle   furniture insect    vegetable animal    building
## [36] clothing  tool      insect    furniture vegetable animal    buildpart
## [43] animal    furniture furniture manmade   building  vehicle   vehicle
## [50] building  manmade   animal    kitchen   buildpart building  kitchen
## [57] insect    vehicle   vegetable animal
## 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         1        0        1         0        0         0      0
##     bodypart       0        0        0         1        0         2      0
##     building       1        0        0         0        0         0      1
##     buildpart      0        0        1         0        0         1      0
##     clothing       0        0        1         0        1         1      0
##     furniture      1        0        1         0        1         0      0
##     insect         1        1        0         0        1         0      0
##     kitchen        0        0        2         0        1         0      1
##     manmade        1        1        0         2        0         0      0
##     tool           1        0        0         1        0         1      0
##     vegetable      0        0        0         0        0         0      1
##     vehicle        1        1        0         1        0         0      0
##              Predicted_Values
## Actual_Values kitchen manmade tool vegetable vehicle
##     animal          0       1    0         0       2
##     bodypart        0       1    0         0       1
##     building        0       1    0         1       1
```

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

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

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

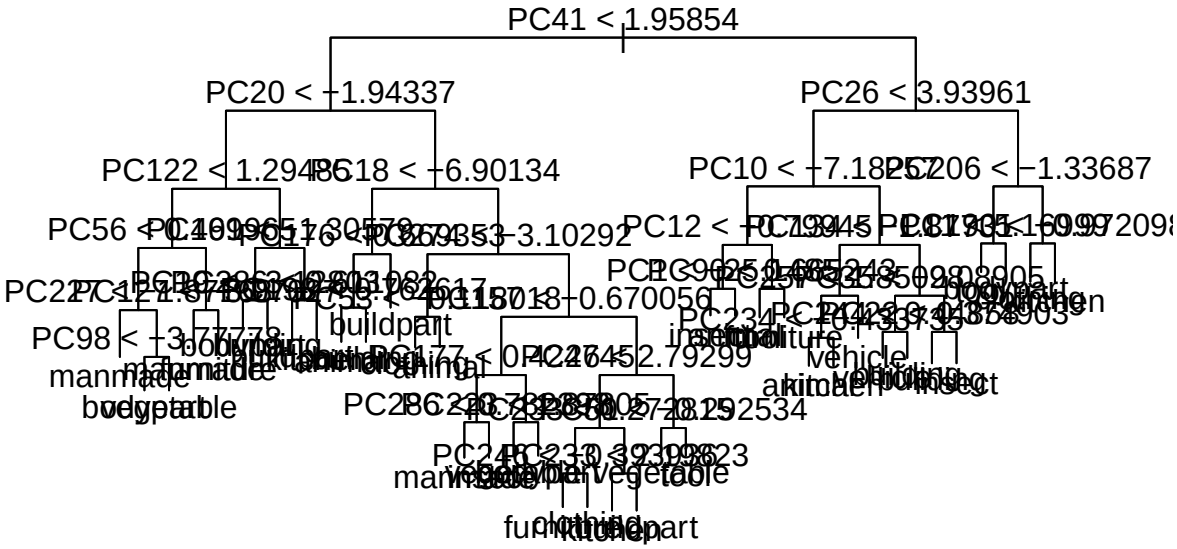
```
## [1] 0.1
```

```
# 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] "PC41"  "PC20"  "PC122" "PC56"  "PC227" "PC98"  "PC127" "PC169" "PC19"
## [10] "PC286" "PC18"  "PC176" "PC198" "PC274" "PC53"  "PC157" "PC177" "PC223"
## [19] "PC26"  "PC233" "PC246" "PC51"  "PC10"  "PC12"  "PC1"   "PC90"  "PC134"
## [28] "PC257" "PC234" "PC35"  "PC244" "PC22"  "PC206" "PC11"  "PC190"
## Number of terminal nodes:  39
## Residual mean deviance:  1.987 = 518.7 / 261
## Misclassification error rate: 0.3867 = 116 / 300
```

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




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

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

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

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

```
## [1] 0.1166667
```

```
# 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         1      2
##     bodypart       1        1        1         0        0         1      0
##     building       0        0        0         0        2         0      0
##     buildpart      0        1        1         1        0         0      0
##     clothing       0        0        2         1        1         1      0
##     furniture      0        0        0         1        0         0      0
##     insect         0        0        1         0        0         0      0
##     kitchen        0        0        0         0        1         2      1
##     manmade        0        1        1         1        0         1      0
##     tool           1        0        0         0        0         0      0
##     vegetable      0        0        0         0        0         0      1
##     vehicle        0        0        0         1        1         0      2
##              rf.pred
## pcs.test.labs kitchen manmade tool vegetable vehicle
##     animal          0       0    0         1       0
##     bodypart        0       0    0         0       1
##     building        0       0    0         2       1
##     buildpart       0       1    0         0       1
##     clothing        0       0    0         0       0
##     furniture       0       2    0         2       0
##     insect          2       1    0         1       0
##     kitchen         0       0    1         0       0
##     manmade         1       0    0         0       0
##     tool            2       0    1         0       1
##     vegetable       0       1    1         1       1
##     vehicle         0       0    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 manmade manmade manmade manmade manmade manmade manmade manmade
## [10] manmade manmade natural manmade manmade manmade natural manmade manmade
## [19] natural manmade manmade manmade manmade manmade manmade manmade manmade
## [28] natural manmade manmade manmade manmade manmade natural manmade manmade
## [37] manmade manmade manmade natural manmade manmade manmade manmade manmade
## [46] manmade manmade manmade manmade manmade manmade manmade manmade manmade
## [55] natural natural manmade manmade natural 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      33       7
##       natural      18       2
```

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

```
## [1] 0.5833333
```

```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      31       9
##       natural      11       9
```

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

```
## [1] 0.6666667
```

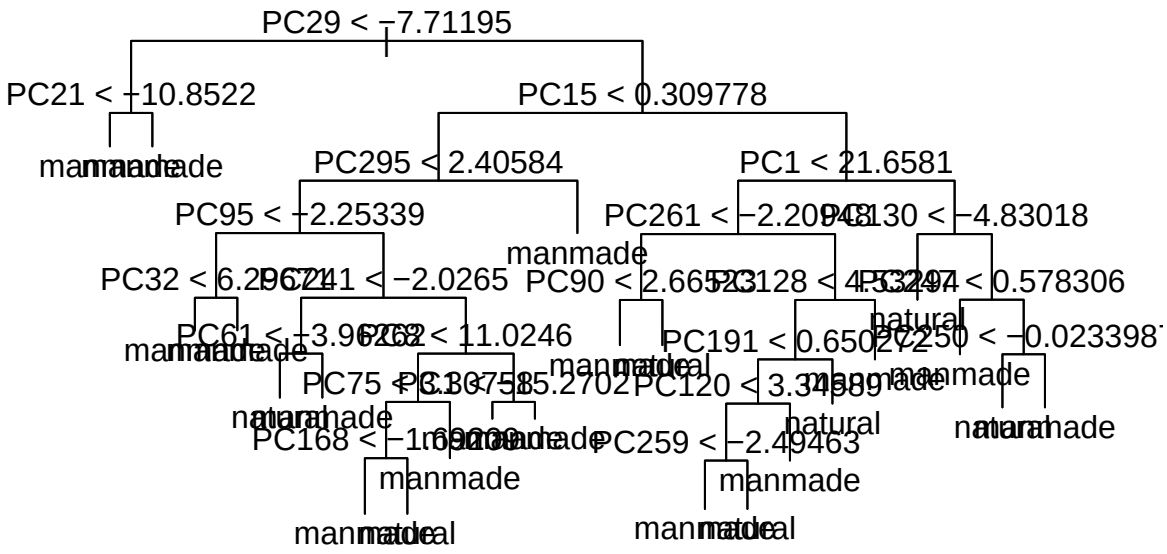
```r
# Decision Trees
set.seed(100)
tree.fit <- tree(as.factor(cls) ~ ., data = pcs.train)
```

```
## 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] "PC29"  "PC21"  "PC15"  "PC295" "PC95"  "PC32"  "PC241" "PC61"  "PC8"
## [10] "PC75"  "PC168" "PC1"   "PC261" "PC90"  "PC128" "PC191" "PC120" "PC259"
## [19] "PC130" "PC247" "PC250"
## Number of terminal nodes:  23
## Residual mean deviance:  0.2867 = 79.41 / 277
## Misclassification error rate: 0.07 = 21 / 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] natural manmade manmade natural manmade manmade manmade manmade manmade
## [10] manmade natural manmade natural manmade manmade manmade manmade manmade
## [19] manmade natural manmade natural manmade manmade natural manmade manmade
## [28] manmade natural natural manmade manmade manmade natural manmade manmade
## [37] manmade manmade natural manmade manmade manmade manmade natural natural
## [46] natural natural manmade manmade natural manmade manmade manmade manmade
## [55] manmade manmade manmade natural natural 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      29      11
##       natural      12       8
```

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

```
## [1] 0.6166667
```

```
# 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
```