Patient4

Anuhya B S

2022-06-08

```
#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-P4.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] 0.08915026 0.15217973 0.19489595 0.22777868 0.25594468 0.27782400
```

```
##   [7] 0.29836968 0.31558151 0.32915646 0.34191489 0.35413252 0.36415893
##  [13] 0.37391547 0.38262426 0.39077335 0.39845659 0.40584138 0.41247832
##  [19] 0.41880630 0.42479259 0.43052354 0.43616517 0.44169411 0.44709526
##  [25] 0.45232193 0.45746654 0.46237500 0.46723870 0.47186012 0.47631255
##  [31] 0.48057104 0.48482663 0.48897871 0.49299920 0.49687172 0.50072176
##  [37] 0.50442271 0.50807328 0.51164370 0.51515696 0.51852736 0.52185221
##  [43] 0.52516761 0.52841861 0.53150300 0.53452945 0.53752313 0.54040382
##  [49] 0.54327040 0.54611484 0.54888562 0.55165552 0.55439278 0.55710231
##  [55] 0.55977162 0.56243979 0.56508870 0.56769184 0.57027629 0.57279136
##  [61] 0.57528955 0.57773152 0.58015277 0.58256551 0.58495463 0.58733299
##  [67] 0.58968642 0.59202908 0.59433659 0.59662578 0.59890331 0.60116662
##  [73] 0.60341368 0.60564425 0.60786242 0.61005471 0.61223791 0.61440993
##  [79] 0.61655983 0.61870406 0.62083321 0.62295126 0.62505470 0.62714625
##  [85] 0.62922528 0.63129197 0.63334416 0.63538418 0.63740403 0.63941211
##  [91] 0.64141500 0.64340925 0.64539545 0.64737307 0.64934444 0.65131373
##  [97] 0.65326485 0.65520637 0.65713686 0.65905894 0.66097772 0.66287917
## [103] 0.66477106 0.66665769 0.66854150 0.67041352 0.67227474 0.67413486
## [109] 0.67598098 0.67782056 0.67965440 0.68148206 0.68330292 0.68511910
## [115] 0.68693036 0.68874034 0.69054103 0.69233283 0.69411536 0.69589524
## [121] 0.69766704 0.69942769 0.70118407 0.70293439 0.70468382 0.70642267
## [127] 0.70815909 0.70988605 0.71160831 0.71332814 0.71503841 0.71674626
## [133] 0.71844676 0.72014166 0.72183394 0.72351766 0.72519798 0.72686924
## [139] 0.72853774 0.73019853 0.73185712 0.73351237 0.73516273 0.73681047
## [145] 0.73845410 0.74009292 0.74172866 0.74336193 0.74498795 0.74661118
## [151] 0.74822884 0.74983923 0.75144811 0.75304838 0.75464540 0.75623422
## [157] 0.75781634 0.75939650 0.76097414 0.76254888 0.76411985 0.76568841
## [163] 0.76725442 0.76881396 0.77036947 0.77191497 0.77345957 0.77499646
## [169] 0.77653078 0.77806329 0.77958772 0.78110528 0.78261966 0.78412892
## [175] 0.78563495 0.78713512 0.78863271 0.79012465 0.79161567 0.79310189
## [181] 0.79458470 0.79606301 0.79753553 0.79900741 0.80047537 0.80194167
## [187] 0.80340439 0.80486544 0.80631727 0.80776640 0.80920998 0.81065040
## [193] 0.81208653 0.81352050 0.81494985 0.81637703 0.81779948 0.81921894
## [199] 0.82063740 0.82205348 0.82346353 0.82486931 0.82627401 0.82767612
## [205] 0.82907532 0.83047049 0.83186102 0.83324795 0.83463183 0.83601102
## [211] 0.83738877 0.83876417 0.84013617 0.84150349 0.84286664 0.84422793
## [217] 0.84558599 0.84694043 0.84828961 0.84963542 0.85097808 0.85231796
## [223] 0.85365051 0.85497992 0.85630871 0.85763644 0.85896209 0.86028435
## [229] 0.86160012 0.86291413 0.86422338 0.86553164 0.86683489 0.86813368
## [235] 0.86942800 0.87072008 0.87200804 0.87329434 0.87457881 0.87585958
## [241] 0.87713499 0.87840968 0.87967889 0.88094479 0.88220864 0.88346812
## [247] 0.88472185 0.88596919 0.88721465 0.88845793 0.88969815 0.89093485
## [253] 0.89216982 0.89339930 0.89462659 0.89585208 0.89707464 0.89829630
## [259] 0.89951321 0.90072682 0.90193777 0.90314676 0.90435265 0.90555348
## [265] 0.90675113 0.90794628 0.90913567 0.91032131 0.91150325 0.91268478
## [271] 0.91386057 0.91503262 0.91620191 0.91736642 0.91852834 0.91969006
## [277] 0.92084919 0.92200322 0.92315552 0.92430595 0.92545146 0.92659295
## [283] 0.92773267 0.92886783 0.93000280 0.93113687 0.93226365 0.93338668
## [289] 0.93450647 0.93562043 0.93673417 0.93784609 0.93895242 0.94005453
## [295] 0.94115506 0.94225117 0.94334242 0.94442827 0.94550892 0.94658920
## [301] 0.94766793 0.94874546 0.94981731 0.95088285 0.95194315 0.95300199
## [307] 0.95405654 0.95510758 0.95615320 0.95719066 0.95822521 0.95925942
## [313] 0.96029101 0.96132014 0.96234090 0.96335979 0.96437179 0.96538186
## [319] 0.96638819 0.96739197 0.96838909 0.96938082 0.97037061 0.97135311
## [325] 0.97232196 0.97327586 0.97420703 0.97512782 0.97604209 0.97694967
```

```
## [331] 0.97784889 0.97873336 0.97961267 0.98047915 0.98134335 0.98219508
## [337] 0.98304126 0.98387569 0.98470236 0.98552656 0.98634100 0.98715208
## [343] 0.98795980 0.98876375 0.98956038 0.99034747 0.99112386 0.99189797
## [349] 0.99266817 0.99342598 0.99418190 0.99493285 0.99568213 0.99642041
## [355] 0.99715427 0.99788041 0.99859722 0.99929976 1.00000000 1.00000000
```

```
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] kitchen   clothing  insect    kitchen   insect    vehicle   furniture
##  [8] kitchen   vehicle   vehicle   tool      insect    insect    buildpart
## [15] manmade   kitchen   manmade   vegetable buildpart manmade   manmade
## [22] vegetable vehicle   insect    vehicle   clothing  clothing  buildpart
## [29] animal    insect    tool      clothing  bodypart  vehicle   clothing
## [36] clothing  tool      vegetable clothing  kitchen   furniture kitchen
## [43] bodypart  vegetable insect    bodypart  vegetable bodypart  insect
## [50] bodypart  vehicle   animal    insect    building  clothing  animal
## [57] kitchen   kitchen   vegetable kitchen
## 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         1        0         0      1
##     bodypart       0        2        0         0        0         0      0
##     building       0        0        0         1        0         1      2
##     buildpart      1        0        0         1        2         0      0
##     clothing       0        0        0         0        2         0      1
##     furniture      0        0        0         0        1         0      1
##     insect         0        0        0         0        1         0      0
##     kitchen        0        1        0         0        2         0      0
##     manmade        0        1        1         0        0         0      0
##     tool           1        0        0         0        0         0      2
```

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

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

```
## [1] 0.1333333
```

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

```
##     vehicle         1       0    0         1       1
```

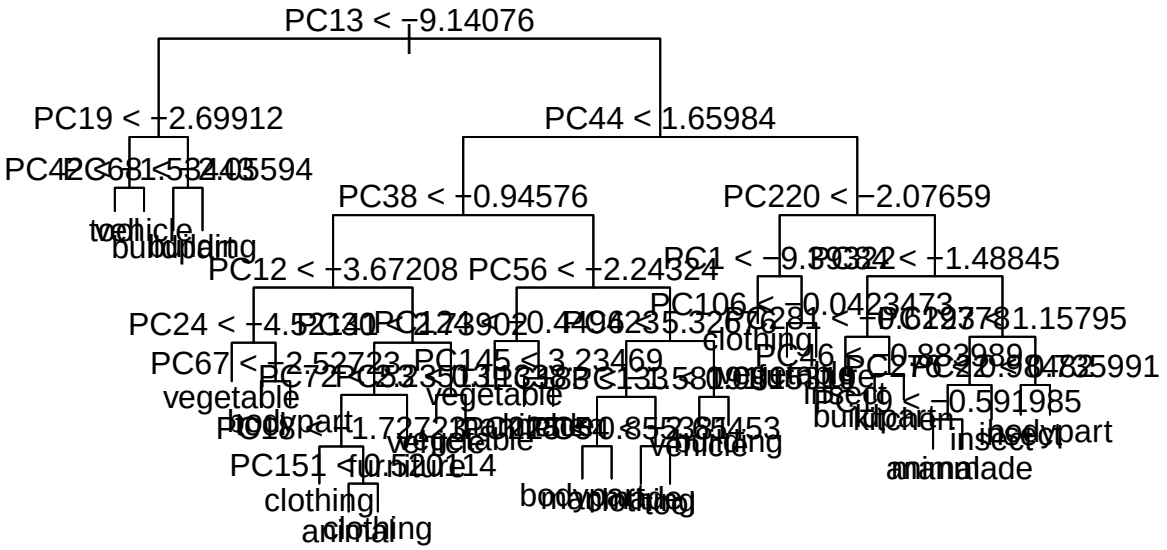
```
print(mean(knn.pred == pcs.test$grp))
```

```
## [1] 0.2166667
```

```
# 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] "PC13"  "PC19"  "PC42"  "PC68"  "PC44"  "PC38"  "PC12"  "PC24"  "PC67"
## [10] "PC30"  "PC72"  "PC18"  "PC151" "PC53"  "PC56"  "PC124" "PC145" "PC6"
## [19] "PC183" "PC225" "PC54"  "PC133" "PC220" "PC1"   "PC106" "PC84"  "PC281"
## [28] "PC46"  "PC223" "PC276" "PC22"
## Number of terminal nodes:  33
## Residual mean deviance:  2.142 = 571.9 / 267
## Misclassification error rate: 0.41 = 123 / 300
```

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




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

```
##  [1] manmade   vegetable building  vegetable animal    manmade   manmade
##  [8] tool      bodypart  vehicle   clothing  tool      tool      bodypart
## [15] manmade   clothing  building  vehicle   buildpart insect    clothing
## [22] manmade   building  building  vegetable tool      clothing  animal
## [29] insect    manmade   furniture bodypart  vegetable building  tool
## [36] buildpart insect    bodypart  tool      vehicle   building  animal
## [43] vegetable vegetable insect    vegetable building  vehicle   building
## [50] building  vegetable clothing  buildpart insect    tool      building
## [57] building  manmade   building  vehicle
## 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        2         0        1         0      0
##     bodypart       1        0        0         0        1         0      0
##     building       0        1        3         0        0         0      0
##     buildpart      0        0        1         1        0         0      0
##     clothing       1        0        0         1        0         1      0
##     furniture      0        0        2         0        0         0      0
##     insect         0        1        0         0        2         0      1
##     kitchen        0        1        1         0        0         0      0
##     manmade        0        0        0         0        0         0      1
##     tool           0        0        2         0        1         0      1
##     vegetable      0        1        0         0        0         0      0
##     vehicle        0        0        1         1        0         0      2
##              tree.pred
## pcs.test.labs kitchen manmade tool vegetable vehicle
##     animal          0       1    0         0       0
##     bodypart        0       0    0         2       1
##     building        0       0    1         0       0
##     buildpart       0       0    2         1       0
##     clothing        0       0    1         0       1
##     furniture       0       1    2         0       0
##     insect          0       1    0         0       0
##     kitchen         0       1    1         1       0
##     manmade         0       1    0         2       1
##     tool            0       0    0         0       1
##     vegetable       0       2    0         2       0
##     vehicle         0       0    0         0       1
```

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

```
## [1] 0.1666667
```

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

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

```
## [1] 0.2333333
```

```
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
```

```
# 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
```

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

```
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      28      12
##       natural      13       7
```

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

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

```
# 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      34       6
##       natural      12       8
```

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

```
## [1] 0.7
```

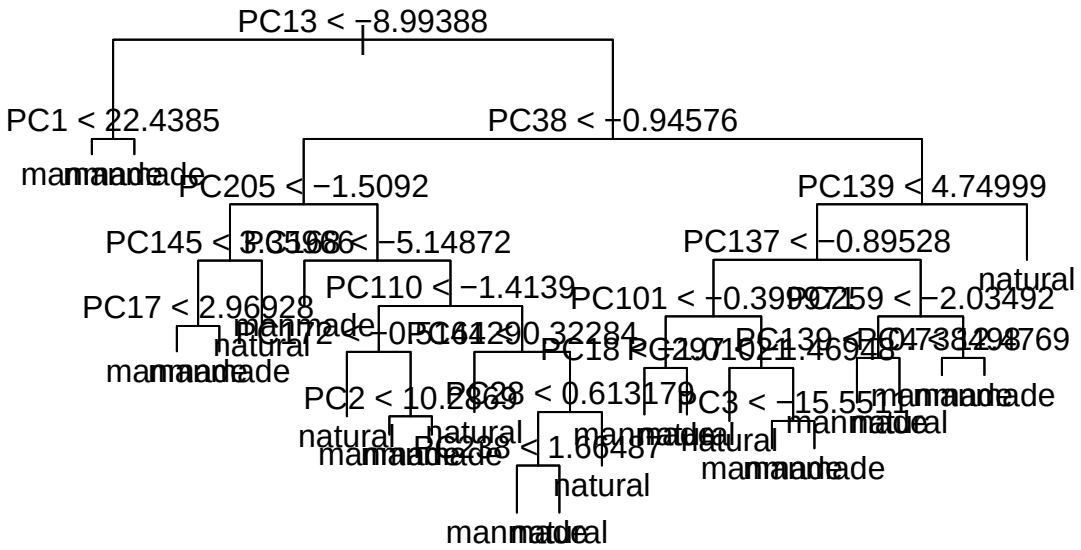
```
# Decision Trees
set.seed(100)
tree.fit <- tree(as.factor(cls) ~ ., data = pcs.train)
```

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

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

```
##
## Classification tree:
## tree(formula = as.factor(cls) ~ ., data = pcs.train)
## Variables actually used in tree construction:
##  [1] "PC13"  "PC1"   "PC38"  "PC205" "PC145" "PC17"  "PC168" "PC110" "PC172"
## [10] "PC2"   "PC41"  "PC28"  "PC238" "PC139" "PC137" "PC101" "PC18"  "PC297"
## [19] "PC3"   "PC259" "PC4"
## Number of terminal nodes:  23
## Residual mean deviance:  0.2703 = 74.87 / 277
## Misclassification error rate: 0.06 = 18 / 300
```

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



```
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 manmade manmade natural natural manmade manmade manmade manmade
## [10] manmade natural manmade manmade natural manmade manmade natural manmade
## [19] manmade manmade natural natural manmade manmade natural manmade manmade
## [28] manmade manmade natural manmade natural manmade manmade natural natural
## [37] manmade natural manmade natural manmade natural manmade manmade manmade
## [46] manmade manmade natural natural manmade manmade natural manmade natural
## [55] manmade manmade manmade manmade manmade manmade
## 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      28      12
##       natural      13       7
```

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

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

```
# 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
```