```r
###決定木(rpart:2進木解析)################################
##wine data
white.wine <- ("https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv")
dat.white <- read.table(white.wine, header = T, sep = ";")
str(dat.white)
colnames(dat.white)
red.wine <- ("https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv")
dat.red <- read.table(red.wine, header = T, sep = ";")
str(dat.red)
colnames(dat.red)
anyNA(dat.white);anyNA(dat.red)
library(caret)
set.seed(123)
dat.w <- createDataPartition( dat.white$quality ,
                              p = 0.75 , list = F )
train.w <- dat.white[ dat.w , ]
test.w <- dat.white[ -dat.w , ]
set.seed(123)
dat.r <- createDataPartition( dat.red$quality ,
                              p = 0.75 , list = F )
train.r <- dat.red[ dat.r , ]
test.r <- dat.red[ -dat.r , ]
#回帰木
##分類木;library(e1071)でチューニング
#minsplit(nodeの最少数),minbucket(leafの最少数,Default;minsplit/3)
#cp(複雑パラメータ)
#maxdepth(木の深さの最大数),xval(number of cross-validations)
options(scipen=50)
library(e1071)
tu.r <- tune.rpart(quality ~ ., data=train.r,
                   cp = c(1e-3,2e-3,3e-3,4e-3,5e-3,6e-3,7e-3,9e-3))
summary(tu.r)
plot(tu.r)
dep.r <- tune.rpart(quality ~ ., data=train.r,
                    maxdepth = 3:10)
summary(dep.r)
plot(dep.r)

#####
library(rpart)
library(partykit)
wr.out0 <- rpart(quality ~ . , data=train.r )
printcp(wr.out0)
sum(residuals(wr.out0)^2)#残差二乗和

#
cotr <- rpart.control(cp = 0.003 , maxdepth = 5 , xval=20)
wr.out <- rpart(quality ~ . , data=train.r ,
                method = "anova",
                parms = list(split = "gini"),#information:エントロピー
                control = cotr)
#method:anova:回帰木,poisson:生起,class:分類木,exp:生存;指定した方がいい
names(wr.out)     #出力内容
as.party(wr.out) #Fitと分割内容
wr.out$cptable
wr.out$variable.importance #重要度
plot(as.party(wr.out))
printcp(wr.out)
plotcp(wr.out)#CP.Graf
sum(residuals(wr.out)^2)#残差二乗和

#回帰推定による残差二乗和と比較
out.lm <- step(lm( quality ~ . ,
               data=train.r))#回帰推定
summary(out.lm)
sum(residuals(out.lm)^2)#回帰推定による残差二乗和
boxplot(data.frame(Rpart=residuals(wr.out),
                   LM=residuals(out.lm)))#残差二乗和の比較
grid()
##Test dataで検証#####################################################
#predict(予測モデル,data)
p.lm.r <- predict(out.lm, test.r)     #重回帰
p.rpart.r <- predict(wr.out, test.r)#回帰木
p.rpart.r0 <- predict(wr.out0, test.r)#回帰木(デフォルト)
#RSME
sqrt( sum((test.r$quality - p.lm.r)^2) / 10788 )   #重回帰
sqrt( sum((test.r$quality - p.rpart.r)^2) / 10788 ) #回帰木
sqrt( sum((test.r$quality - p.rpart.r0)^2) / 10788 ) #回帰木(デフォルト)
#MAE
mean(abs(test.r$quality - p.lm.r))     #重回帰
mean(abs(test.r$quality - p.rpart.r)) #回帰木
mean(abs(test.r$quality - p.rpart.r0)) #回帰木(デフォルト)
#cor
cor(p.lm.r , test.r$quality)       #重回帰
cor(p.rpart.r , test.r$quality)    #回帰木
cor(p.rpart.r0 , test.r$quality)    #回帰木(デフォルト)
##分類木#############################################################
#第1列から48列:変数名に用いた文字列がメールに使用された頻度
#num857のようにnum***はその数値***が現れた頻度
#49列から54列:記号;、(、[、!、$、#の使用頻度
```

```r
#55列から57列：用いられた大文字の平均、大文字が連続使用された
#最も長い文字列の文字数、用いられた大文字の総数
##train.test data作成
library(kernlab);data(spam)
library(caret)
anyNA(spam)
set.seed(123)
dat.s <- createDataPartition( spam$type , p = 0.7 , list = F )
train.spam <- spam[ dat.s ,]
test.spam  <- spam[-dat.s ,]
##分類木;library(e1071)でチューニング
#minsplit(nodeの最少数),minbucket(leafの最少数,Default;minsplit/3)
#cp(複雑パラメータ)
#maxdepth(木の深さの最大数),xval(number of cross-validations)
options(scipen=50)
library(e1071)
tun <- tune.rpart(type ~ ., data=train.spam,
                  cp = c(1e-4,1e-3,3e-3,5e-3,7e-3,1e-2))
summary(tun)
plot(tun)

msp <- tune.rpart(type ~ ., data=train.spam,
                  minsplit=seq(10,50,10))
summary(msp)
plot(msp)

dep <- tune.rpart(type ~ ., data=train.spam,
                  maxdepth = seq(3,10,1))
summary(dep)
plot(dep)

#####
library(rpart)
library(partykit)
cont <- rpart.control(minsplit = 30, minbucket = round(30/3),#minsplit/3
                      cp = 0.001 , maxdepth = 6 ,xval=20)
spam.out <- rpart(type ~ ., data=train.spam , method = "class",
                  parms = list(split = "gini"),#information:エントロピー
                  control = cont)
names(spam.out)    #出力内容
as.party(spam.out) #Fitと分割内容
spam.out$cptable
spam.out$variable.importance #重要度
plot(as.party(spam.out))
pd <- predict(spam.out , type = "class")
tb <- xtabs( ~ type + pd , data = train.spam )
tb
round((tb[1,1] + tb[2,2])/length(train.spam$type),3)*100
#confusion matrix
pd <- predict(spam.out , type = "class")
library(caret)
confusionMatrix(pd , train.spam$type )
##Test dataで検証#################################################
#predict(予測モデル,data)
p.rpart.t <- predict(spam.out, test.spam , type = "class")
tb.t <- xtabs( ~ type + p.rpart.t , data = test.spam )
tb.t
round((tb.t[1,1] + tb.t[2,2])/length(test.spam$type),3)*100
library(caret)
confusionMatrix(p.rpart.t, test.spam$type )
##判別分析#######
library(MASS)
lda.out <- lda(type ~ ., data=train.spam )
summary(lda.out)
predict.lda <- predict(lda.out)
tab <- table(train.spam$type , predict.lda$class)
round(((tab[1,1]+ tab[2,2])/sum(tab)*100),2)
predict.out.test <- predict(lda.out , newdata = test.spam)
tab.test <- table(test.spam$type , predict.out.test$class)
#rn.ics
round(((tab.test[1,1] + tab.test[2,2])/sum(tab.test)*100),2) #判別
round((tb.t[1,1] + tb.t[2,2])/length(test.spam$type),3)*100  #分類木

##多クラス分類
##手書き数 data
test.su <- ("https://archive.ics.uci.edu/ml/machine-learning-databases/optdigits//optdigits.tes")
test.su <- read.table(test.su  , sep = ",")
train.su <- ("https://archive.ics.uci.edu/ml/machine-learning-databases/optdigits//optdigits.tra")
train.su <- read.table(train.su , sep = ",")
str(train.su)
colnames(train.su)
str(test.su)
colnames(test.su)
table(test.su$V65)
table(train.su$V65)
#y:factor変換 confusion matrix出力のため(factor変換:解析には不要)
train.su$V65 <- factor(train.su$V65)
str(train.su)
test.su$V65 <- factor(test.su$V65)
str(test.su)
```

```r
colnames(train.su)
colnames(test.su)
table(test.su$V65)
table(train.su$V65)
##
##分類木;library(e1071)でチューニング
options(scipen=50)
library(e1071)
tue <- tune.rpart(V65 ~ ., data=train.su ,
                  cp = c(1e-4,4e-4,7e-4,1e-3,3e-3,5e-3,7e-3,1e-2))
summary(tue)
plot(tue)
readline(tue)
spl <- tune.rpart(V65 ~ ., data=train.su ,
                  minsplit=seq(10,80,20))
summary(spl)
plot(spl)
readline(spl)
mdp <- tune.rpart(V65 ~ ., data=train.su ,
                  maxdepth = 3:10)
summary(mdp)
plot(mdp)
readline(mdp)
#####
library(rpart)
library(partykit)
cotr <- rpart.control(minsplit = 10, minbucket = round(10/3),#minsplit/3
                      cp = 0.0004 , maxdepth = 8 ,xval=20)
suchi.out <- rpart(V65 ~ ., data=train.su , method = "class",
                   parms = list(split = "gini"),#information:エントロピー
                   control = cotr)
suchi.out
names(suchi.out)     #出力内容
as.party(suchi.out) #Fitと分割内容
suchi.out$cptable
suchi.out$variable.importance #重要度
plot(as.party(suchi.out))
printcp(suchi.out)
plotcp(suchi.out)#CP.Graf
#クロス集計:的中率
est.su <- predict(suchi.out , type = "class")
est.su2 <- xtabs( ~ V65 + est.su , data = train.su )
est.su2
#round(sum(diag(est.su2))/sum(table(train.su$V65)),4)*100
round(sum(diag(est.su2)) /sum(est.su2),4)*100
#confusion matrix
library(caret)
confusionMatrix(est.su, train.su$V65 )
##Test dataで検証#####################################################
#predict(予測モデル,data)
p.rpart.su <- predict(suchi.out, test.su , type = "class")
est.su2.2 <- xtabs( ~ V65 + p.rpart.su , data = test.su )
est.su2.2
round(sum(diag(est.su2.2)) /sum(est.su2.2),4)*100
#confusion matrix
library(caret)
confusionMatrix(p.rpart.su, test.su$V65 )
#
options(scipen=0)

```