```r
##K近傍法(K-NN)###########################################
library(kernlab);data(spam)
anyNA(spam)
str(spam)
#class(spam$type)#classの確認
colnames(spam)
library(caret)
set.seed(123)
dat.s <- createDataPartition( spam$type , p = 0.75 , list = F )
train.spam <- spam[ dat.s ,]
test.spam  <- spam[-dat.s ,]
#knn:k近傍法は距離の近いk個のラベルを用いて分類する.距離自体は分類に
#用いない,近いデータを探すのに用いるだけ.
#e1071()によるチューニング
library(e1071)
##knn
set.seed(123)
knnfit <- tune.knn(x = train.spam[,-58], y = train.spam[,58], k = 1:20,
                   tunecontrol=tune.control(sampling="cross"),
                   cross=5)#boot,fix:回数
summary(knnfit)
plot(knnfit)
library(class)
set.seed(123)
knn.fit <- knn(train = train.spam[,-58] , test = test.spam[,-58] ,
               cl = train.spam[,58] , prob = T , k = 1 )
tt <- table(test.spam$type, knn.fit)
round((tt[1,1]+tt[2,2])/length(test.spam$type),4)*100
#confusion matrix
library(caret)
confusionMatrix(knn.fit, test.spam$type)
##重みづけk近傍法kknn:距離が近い方が類似性が高いと考える
#ミンコフスキー距離を用いて重みづけ
##kknn:チューニング
library(kknn)
train1 <- train.kknn(type ~ ., data = train.spam,kmax = 20,
                     kernel = c("rectangular", "triangular",
                                "epanechnikov", "gaussian",
                                "rank", "optimal"),
                     distance = 1 , scale = T )#ミンコフスキー定数
plot(train1)
train1
train2 <- train.kknn(type ~ ., data = train.spam,kmax = 20,
                     kernel = c("rectangular", "triangular",
                                "epanechnikov", "gaussian",
                                "rank", "optimal"),
                     distance = 2 , scale = T )#ミンコフスキー定数
plot(train2)
train2

##kknn
library(kknn)
set.seed(123)
kknn.out <- kknn(type ~ ., train.spam, test.spam,
                 k = 15 ,kernel="triangular", distance = 2 , scale = T)
t <- table(test.spam$type, kknn.out$fit)
t
round((t[1,1]+t[2,2])/length(test.spam$type),4)*100
library(gmodels)
CrossTable(x = test.spam$type , y = kknn.out$fit ,prop.chisq = F )
#confusion matrix
library(caret)
confusionMatrix(kknn.out$fit, test.spam$type)
##多クラス分類
##手書き数 data
test.su <- ("http://archive.ics.uci.edu/ml/machine-learning-databases/optdigits//optdigits.tes")
test.su <- read.table(test.su  , sep = ",")
train.su <- ("http://archive.ics.uci.edu/ml/machine-learning-databases/optdigits//optdigits.tra")
train.su <- read.table(train.su , sep = ",")
str(train.su)
str(test.su)
#factor変換
train.su$V65 <- factor(train.su$V65)
str(train.su)
test.su$V65 <- factor(test.su$V65)
str(test.su)
colnames(train.su)
colnames(test.su)
table(test.su$V65)
table(train.su$V65)
#Z変換
train.su.z <- scale(train.su[,c(1:64)])
train.su.z <- cbind(data.frame(train.su.z) , train.su$V65)
names(train.su.z)[65] <- c("su")
str(train.su.z)
test.su.z <- scale(test.su[,c(1:64)])
test.su.z <- cbind(data.frame(test.su.z) , test.su$V65)
names(test.su.z)[65] <- c("su")
str(test.su.z)
#normalize変換
```

```r
norm <- function(x){
    return((x-min(x))/(max(x)-min(x)))
          }
train.su.norm <- data.frame(lapply(train.su[1:64] , norm))
train.su.norm <- cbind(data.frame(train.su.norm) , train.su$V65)
names(train.su.norm)[65] <- c("su")
str(train.su.norm)
anyNA(train.su.norm[,c(2:30,41:64)])
test.su.norm <- data.frame(lapply(test.su[1:64] , norm))
test.su.norm <- cbind(data.frame(test.su.norm) , test.su$V65)
names(test.su.norm)[65] <- c("su")
str(test.su.norm)
anyNA(test.su.norm[,c(2:30,41:64)])

##knn
set.seed(123)
knnfit.m <- tune.knn(x = train.su[,-65], y = train.su[,65]
                     , tunecontrol = tune.control(sampling = "boot"),
                     k = 1:11)
summary(knnfit.m)
plot(knnfit.m)
library(class)
set.seed(123)
knn.fitm <- knn(train = train.su[,-65] , test = test.su[,-65] ,
                cl = train.su[,65] , prob = T , k = 1 )
tt.m <- table(test.su$V65, knn.fitm)
round(sum(diag(tt.m))/length(test.su$V65),4)*100
#confusion matrix
library(caret)
confusionMatrix(knn.fitm, test.su$V65)
#knn.z
set.seed(123)
knnfit.z <- tune.knn(x = train.su.z[c(2:30,41:64)], y = train.su.z[,65]
                     , tunecontrol = tune.control(sampling = "boot"),
                     k = 1:11)
summary(knnfit.z)
plot(knnfit.z)
library(class)
set.seed(123)
knn.fitm.z <- knn(train = train.su.z[c(2:30,41:64)] ,
                  test = test.su.z[c(2:30,41:64)] ,
                  cl = train.su.z[,65] , prob = T , k = 1 )
tt.m.z <- table(test.su.z$su, knn.fitm.z)
round(sum(diag(tt.m.z))/length(test.su.z$su),4)*100
#confusion matrix
library(caret)
confusionMatrix(knn.fitm.z, test.su.z$su)
###kknn
library(kknn)
train.m <- train.kknn(V65 ~ ., data = train.su,kmax = 25,
                      kernel = c("rectangular", "triangular",
                                 "epanechnikov", "gaussian",
                                 "rank", "optimal"),
                      distance = 2 , scale = T )#ミンコフスキー定数
plot(train.m)
train.m
##kknn
library(kknn)
set.seed(123)
kknn.outm <- kknn(V65 ~ ., train.su, test.su,
                  k = 4 ,kernel="triangular", distance = 2 , scale = T)
tm <- table(test.su$V65, kknn.outm$fit)
round(sum(diag(tm)) /sum(tm),4)*100
#library(gmodels)
#CrossTable(x = test.su$class , y = kknn.out$fit ,prop.chisq = F )
#confusion matrix
library(caret)
confusionMatrix(kknn.outm$fit, test.su$V65)

```