HW13

Jackie Liang

December 10, 2018

```
library(dplyr)

## Warning: package 'dplyr' was built under R version 3.4.4

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(caret)

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.4.4

library(nnet)

if (!exists("mtrain")) {
  mtrain <- read.csv("mnist_train.csv", header=F) %>% as.matrix
  train_classification <- mtrain[,1]
  y <- factor(train_classification, levels=c(0,1))
  mtrain <- mtrain[,-1]/256 #this is the x matirix

  colnames(mtrain) <- 1:(28^2)

  x <- mtrain[1:1000,]

}


for (i in 1:length(train_classification)){
  cn <- train_classification[i]
  if (cn==3){
    cn <- 1
```

```
  } else {
    cn <- 0
  }
  y[i] <- cn
}

y <- factor(y, levels = c(0,1))
y <- y[1:1000]
```

caret 0 decay

```
tuning_df3 <- data.frame(size=5, decay=0)
tuning_df1 <- data.frame(size=1, decay=0)
tuning_df2 <- data.frame(size=3, decay=0)

fitControl <- trainControl(## 2-fold CV
  method = "repeatedcv",
  number = 2,
  repeats = 2)

t_out1 <- caret::train(x=x, y=y, method="nnet",
                       trControl = fitControl,
                       tuneGrid=tuning_df1, maxit=1000, MaxNWts=10000)

t_out2 <- caret::train(x=x, y=y, method="nnet",
                        trControl = fitControl,
                        tuneGrid=tuning_df2, maxit=1000, MaxNWts=10000)

t_out3 <- caret::train(x=x, y=y, method="nnet",
                        trControl = fitControl,
                        tuneGrid=tuning_df3, maxit=1000, MaxNWts=10000)

y_fit3 <- predict(t_out3, x)
y_fit1 <- predict(t_out1, x)
y_fit2 <- predict(t_out2, x)

true_y <- y

#model 1
n_samples <- nrow(x)
error1 <- sum(true_y != y_fit1)/n_samples
pred_error1 <- error1

pred_error1
```

```r
error2 <- sum(true_y != y_fit2)/n_samples
pred_error2 <- error2

pred_error2

error3 <- sum(true_y != y_fit3)/n_samples
pred_error3 <- error3

pred_error3

#Models with decay

tuning1 <- data.frame(size=3, decay=0)
tuning2 <- data.frame(size=3, decay=.5)
tuning3 <- data.frame(size=3, decay=1)


model1 <- caret::train(x=x, y=y, method="nnet",
                       trControl = fitControl,
                       tuneGrid=tuning1, maxit=1000, MaxNWts=10000)
model2 <- caret::train(x=x, y=y, method="nnet",
                       trControl = fitControl,
                       tuneGrid=tuning2, maxit=1000, MaxNWts=10000)
model3 <- caret::train(x=x, y=y, method="nnet",
                       trControl = fitControl,
                       tuneGrid=tuning3, maxit=1000, MaxNWts=10000)

error4 <- sum(true_y != model1)/n_samples
pred_error4 <- error4
pred_error4

## [1] 1

error5 <- sum(true_y != model2)/n_samples
pred_error5 <- error5
pred_error5

## [1] 1

error6 <- sum(true_y != model3)/n_samples
pred_error6 <- error6
pred_error6

## [1] 1
```

Checking against mnist_test

```
if (!exists("mtrain2")) {
  mtrain2 <- read.csv("mnist_test.csv", header=F) %>% as.matrix
  train_classification2 <- mtrain2[,1]
  mtrain2 <- mtrain2[,-1]/256 #x values


  colnames(mtrain2) <- 1:(28^2)
  rownames(mtrain2) <- NULL

  x2 <- mtrain2[1:1000,]
}

y2 <- rep(NA, length(train_classification))
#Converting all threes to one and all other numbers to zero
for (i in 1:length(train_classification2)){
  cn <- train_classification2[i]
  if (cn==3){
    cn <- 1
  } else {
    cn <- 0
  }
  y2[i] <- cn
}

y2 <- factor(y, levels=c(0,1))
y2 <- y[1:1000]

true_y2 <- y2
pred_y2 <- predict(t_out1, x2)
n_samples2 <- nrow(x2)

error_a <- sum(true_y2 != pred_y2)/n_samples2
pred_error_a <- error_a

pred_error_a

## [1] 0.19
```