Classification

Stefan Haböck & Bernhard Mayr

30 11 2021

## Dependencies

library(dplyr)  
library(lolR)  
library(class)  
library(e1071)

printModelResults <- function(predict, actual\_labels){  
 precision <- sum(predict & actual\_labels) / sum(predict)  
 recall <- sum(predict & actual\_labels) / sum(actual\_labels)  
 fmeasure <- 2 \* precision \* recall / (precision + recall)  
  
 cat('precision: ')  
 cat(precision \* 100)  
 cat('%')  
 cat('\n')  
  
 cat('recall: ')  
 cat(recall \* 100)  
 cat('%')  
 cat('\n')  
  
 cat('f-measure: ')  
 cat(fmeasure \* 100)  
 cat('%')  
 cat('\n')  
}

data.ctd <- read.table("./CtD.csv", sep=";", header=TRUE, strip.white=TRUE) %>%  
 tibble::as\_tibble()  
data.ttd <- read.table("./ttd.csv", sep=",", header=FALSE, strip.white=TRUE) %>%  
 tibble::as\_tibble()  
  
names(data.ctd) <- c("lectureName", "isTechnology", "isInterdisciplinary", "isMedia", "isBiology")  
  
data.raw <- bind\_cols(data.ctd, data.ttd)

# Test Train Split  
data.splitFactor = .7 # 70% Training Data / 30% Test Data  
data.randomization = sort(sample(nrow(data.raw), nrow(data.raw) \* data.splitFactor))  
data.train <- data.raw[data.randomization,]  
data.test <- data.raw[-data.randomization,]  
  
# Training Data  
train.features = data.train %>% select(6:ncol(data.train))  
train.labels = data.train$isTechnology  
  
# Test Data  
test.data = data.test %>% select(6:ncol(data.test))  
test.expected = data.test$isTechnology

model.rocchio <- lol.classify.nearestCentroid(train.features, train.labels)  
result.rocchio <- predict(model.rocchio, test.data)  
  
printModelResults(result.rocchio, test.expected)

## precision: 88%  
## recall: 100%  
## f-measure: 93.61702%

result.knn3 <- knn(train.features, test.data, cl=train.labels, k=3)  
printModelResults(as.numeric(as.character(result.knn3)), test.expected)

## precision: 70.37037%  
## recall: 86.36364%  
## f-measure: 77.55102%

result.knn5 <- knn(train.features, test.data, cl=train.labels, k=5)  
printModelResults(as.numeric(as.character(result.knn5)), test.expected)

## precision: 72.41379%  
## recall: 95.45455%  
## f-measure: 82.35294%

result.knn7 <- knn(train.features, test.data, cl=train.labels, k=7)  
printModelResults(as.numeric(as.character(result.knn7)), test.expected)

## precision: 65.625%  
## recall: 95.45455%  
## f-measure: 77.77778%

data.temp <- data.train %>% select(isTechnology, 6:ncol(data.train))  
model.naivebayes <- naiveBayes(as.factor(isTechnology) ~ ., data=data.temp)  
  
model.naivebayes <- naiveBayes(train.features, as.factor(train.labels))  
result.naivebayes <- predict(model.naivebayes, test.data)  
result.naivebayes

## [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## Levels: 0 1

printModelResults(as.numeric(as.character(result.naivebayes)), test.expected)

## precision: 61.11111%  
## recall: 100%  
## f-measure: 75.86207%

## Summery