Spectral clustering - raport

Przemysław Kaleta

January 25, 2019

Zapisanie wyników benchmarkowych

Poniżej kod skryptu, w którym przetestowałem algorytmy na wszystkich zbiorach benchamrkowych i zapisałem wyniki do pliku `results.csv`.

```
library(mclust)
library(dendextend)
library(genie)
library(stringi)

source("spectral.R")


read_data <- function(benchmark, dataset){
  matrix_file_name <- paste(dataset, ".data.gz", sep="")
  labels_file_name <- paste(dataset, ".labels0.gz", sep="")
  matrix_path <- file.path("..", "benchmarks", benchmark, matrix_file_name)
  labels_path <- file.path("..", "benchmarks", benchmark, labels_file_name)
  X <- as.matrix(read.table(matrix_path))
  Y <- as.matrix(read.table(labels_path))
  return(list(X=X, Y=Y))
}

plot_data <- function(X, Y, title=""){
  plot(X[, 1], X[, 2], col=unlist(Y), pch=20)
  title(title)
}

result_spectral <- function(benchmark, dataset, M=20, k=NULL, scale=FALSE, plot=FALSE){
  data <- read_data(benchmark, dataset)
  X <- data$X
  if(scale){
    X <- scale(X)
  }
  Y <- data$Y
  if(is.null(k)){
    k <- length(unique(unlist(Y)))
  }
  set.seed(42)  # because kmeans in spectral clustering randomly initializes centers
  Y_pred <- spectral_clustering(X, k, M)
  if(plot){
    plot_data(X, Y_pred, paste(paste(benchmark, dataset, sep="/"), ": spectral ", sep=""))
  }
  algorithm <- "spectral"

  return(list(benchmark=benchmark,
              dataset=dataset,
```

```
              algorithm=algorithm,
              FM=FM_index(Y, Y_pred),
              AR=adjustedRandIndex(Y, Y_pred)))
}


result_hclust <- function(benchmark, dataset, method="complete", k=NULL, scale=FALSE, plot=FALSE){
  data <- read_data(benchmark, dataset)
  X <- data$X
  if(scale){
    X <- scale(X)
  }
  Y <- data$Y
  if(is.null(k)){
    k = length(unique(unlist(Y)))
  }

  hc <- hclust(dist(X), method)
  Y_pred <- cutree(hc, k=k)

  if(plot){
    plot_data(X, Y_pred, paste(paste(benchmark, dataset, sep="/"), ": hclust ", method, sep=""))
  }

  algorithm <- paste("hclust", method, sep="_")
  return(list(benchmark=benchmark,
              dataset=dataset,
              algorithm=algorithm,
              FM=FM_index(Y, Y_pred),
              AR=adjustedRandIndex(Y, Y_pred)))
}


result_genie <- function(benchmark, dataset, k=NULL, scale=FALSE){
  data <- read_data(benchmark, dataset)
  X <- data$X
  if(scale){
    X <- scale(X)
  }
  Y <- data$Y
  if(is.null(k)){
    k = length(unique(unlist(Y)))
  }

  hc <- hclust2(dist(X))
  Y_pred <- cutree(hc, k=k)

  algorithm <- "genie"

  return(list(benchmark=benchmark,
              dataset=dataset,
              algorithm=algorithm,
              FM=FM_index(Y, Y_pred),
```

```
              AR=adjustedRandIndex(Y, Y_pred)))
}

result <- list()
benchmarks <- c("fcps", "graves", "other", "sipu", "wut")
for(benchmark in benchmarks){
  matrix_ending <- "data.gz"
  labels_ending <- "labels0.gz"
  benchmark_path <- file.path("..", "benchmarks", benchmark)
  datasets <- list.files(benchmark_path)
  for(dataset in datasets){
    if(endsWith(dataset, ".txt")){
      dataset <- stri_sub(dataset, 0, -5)
      print(paste("Currently processing", benchmark, dataset))
      data <- read_data(benchmark, dataset)
      X <- data$X
      Y <- data$Y

      # testing hclust methods
      hclust_methods <- c("complete", "average", "mcquitty", "median", "centroid")
      for(method in hclust_methods){
        result <- rbind(result, result_hclust(benchmark, dataset, method=method))
      }

      # testing other methods
      result <- rbind(result,
                      result_genie(benchmark, dataset),
                      result_spectral(benchmark, dataset))
    }
  }
  write.csv2(result, "results.csv")  # writing to have at least partial results
}

write.csv(result, "results.csv")
```