Lehmann's alternative on Digits dataset

2023-05-17

We consider Lehmann's alternative with $k = 2$ and we show that the theoretical result that closed testing procedure with Wilcoxon-Mann-Whitney local test is the Locally Most Powerful Invariant test for the global null is validated also by numerical simulations.

```
library(doSNOW)
library(foreach)
library(nout)
library(tictoc)
library(isotree)
library(readr)
library(R.matlab)



compact_results = function(res){
  resT=as.data.frame(t(res))
  discoveries = as.data.frame(cbind("d_BH"=unlist(resT$d_BH),
                                    "d_StoBH"=unlist(resT$d_StoBH),
                                    "d_Sim"=unlist(resT$d_Sim),
                                    "d_StoSimes"=unlist(resT$d_StoSimes),
                                    "d_WMW"=unlist(resT$d_WMW)))
  mean.discoveries = apply(discoveries, MARGIN = 2, FUN = mean)
  power.GlobalNull = as.data.frame(discoveries>0)
  mean.powerGlobalNull = apply(power.GlobalNull, MARGIN = 2, FUN = mean)
  return(list("discoveries" = discoveries,
              "mean.discoveries" = mean.discoveries,
              "power.GlobalNull" = power.GlobalNull,
              "mean.powerGlobalNull" = mean.powerGlobalNull,
              "pi.not" = unlist(resT$pi.not),
              "uniques"=unlist(resT$uniques),
              "n1"=unlist(resT$n1),
              "alpha"=unlist(resT$alpha)))
}
```

Digits dataset

```
data = readMat("G:\\Il mio Drive\\PHD\\Progetto di ricerca\\Conformal Inference Project\\Simulazioni\\7
# data = readMat("~/nout/trials/RealData/Datasets/Dataset digits/pendigits.mat")

dataset = cbind(data$X, data$y); colnames(dataset)[ncol(dataset)] = "y"
in_ind = which(dataset[,ncol(dataset)]==0)
out_ind = which(dataset[,ncol(dataset)]==1)

# Initializing parameters
set.seed(321)
```

```
B=10^4

l = 1683
m = 249
n = 50
k=2 # exponent of Lehmann's alternative
myalpha = n/(m+1)


tr_ind = sample(in_ind, size = l)
in_ind2 = setdiff(in_ind, tr_ind)
tr = dataset[tr_ind,]
n_cpus = parallel::detectCores()
iso.fo = isotree::isolation.forest(tr, ndim = ncol(dataset), ntrees = 180, sample_size = 256,
                                   nthreads = n_cpus, scoring_metric = "depth",
                                   output_score = TRUE)
isofo.model = iso.fo$model
mycrit = nout::critWMW(m = n, n = m, alpha = myalpha)
```

Proportion of outliers equal to 0.05

```
n1=round(0.05*n)

cl <- makeCluster(parallel::detectCores())
clusterEvalQ(cl, {library(isotree)})
```

```
## [[1]]
## [1] "isotree"   "snow"      "stats"     "graphics"  "grDevices" "utils"
## [7] "datasets"  "methods"   "base"
##
## [[2]]
## [1] "isotree"   "snow"      "stats"     "graphics"  "grDevices" "utils"
## [7] "datasets"  "methods"   "base"
##
## [[3]]
## [1] "isotree"   "snow"      "stats"     "graphics"  "grDevices" "utils"
## [7] "datasets"  "methods"   "base"
##
## [[4]]
## [1] "isotree"   "snow"      "stats"     "graphics"  "grDevices" "utils"
## [7] "datasets"  "methods"   "base"
```

```
registerDoSNOW(cl)

res = foreach(b = 1:B, .combine=cbind) %dopar% {
  n0 = n - n1
  N = n0 + m
  in_index3 = sample(in_ind, size = N)
  cal_ind = in_index3[1:m]
  tein_ind = in_index3[(m + 1):N]
  teout_ind = sample(out_ind, size = n1*k)
  cal = dataset[cal_ind,]
  te = dataset[c(tein_ind, teout_ind),]
```

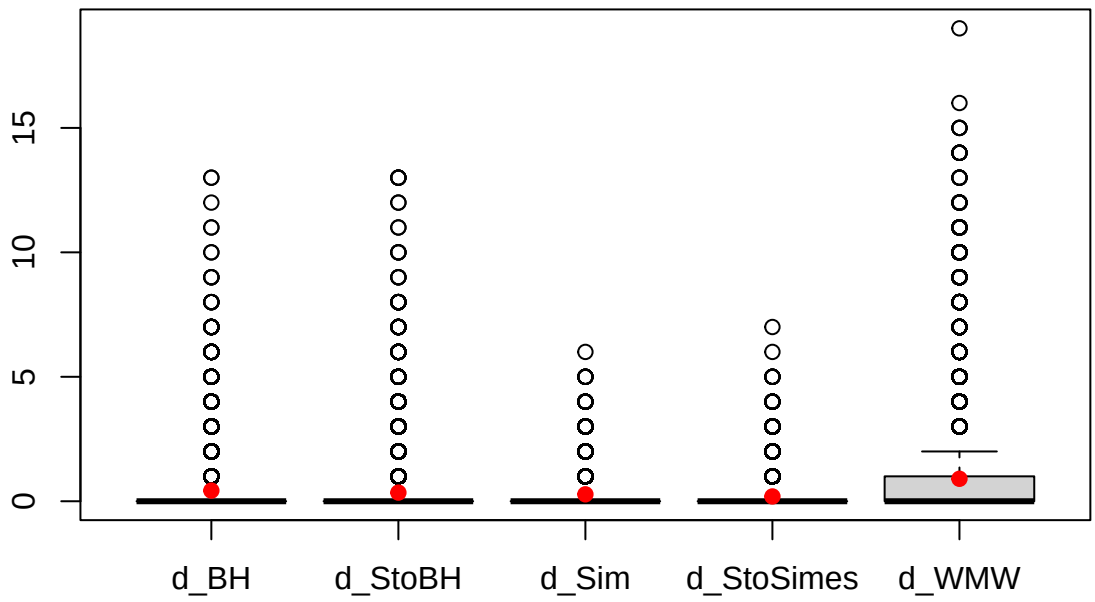
```
  S_cal = predict.isolation_forest(isofo.model, cal, type = "score")
  augmented.S_te = predict.isolation_forest(isofo.model, te, type = "score")
  S_te = c(augmented.S_te[1:n0],
           sapply(0:(n1-1), FUN=function(i) max(augmented.S_te[1+k*i], augmented.S_te[k+k*i])))
  d_WMW = nout::d_mannwhitney(S_Y = S_te, S_X = S_cal, crit = mycrit)
  d_Sim = nout::d_Simes(S_X = S_cal, S_Y = S_te, alpha = myalpha)
  StoSimes = nout::d_StoreySimes(S_X = S_cal, S_Y = S_te, alpha = myalpha)
  d_StoSimes = StoSimes$d
  pi.not = StoSimes$pi.not
  d_BH = nout::d_benjhoch(S_X = S_cal, S_Y = S_te, alpha = myalpha)
  d_StoBH = nout::d_StoreyBH(S_X = S_cal, S_Y = S_te, alpha = myalpha)
  uniques = length(unique(c(S_cal, S_te)))
  return(list("d_BH" = d_BH,
              "d_StoBH" = d_StoBH,
              "d_Sim" = d_Sim,
              "d_StoSimes" = d_StoSimes,
              "d_WMW" = d_WMW,
              "uniques" = uniques,
              "n1" = n1,
              "pi.not" = pi.not,
              "alpha" = myalpha))

}

stopCluster(cl)

results = compact_results(res)

boxplot(results$discoveries, main="Digits | Distribution of the number of discoveries")
points(x=1:5, y=results$mean.discoveries, pch=19, col="red")
```




```
results$mean.discoveries
```

```
##       d_BH    d_StoBH      d_Sim d_StoSimes      d_WMW
##     0.4279     0.3470     0.2778     0.1929     0.9026
```

```
results$mean.powerGlobalNull
```

```
##       d_BH    d_StoBH      d_Sim d_StoSimes      d_WMW
##     0.2059     0.1388     0.2059     0.1388     0.2534
```

```
resDigits005 = results
save(resDigits005,
     file="~/nout/trials/RealData/PowerStudy/New!/alpha0.2/DigitsOnly0.2/Lehmann2/resDigits005")
```