Credit Card dataset Power Analysis

Significance level 0.2

2023-05-14

```
library(nout)
library(R.matlab)
library(isotree)
library(tictoc)
```

Credit Card Fraud Detection dataset

The aim is to compare on Credit Card dataset (available at https://www.kaggle.com/mlg-ulb/creditcardfraud) the performance of three closed testing procedures, which respectively use Simes local test with and without Storey estimator for the proportion of true null hypotheses and Wilcoxon-Mann-Whitney local test.

Credit card dataset consists of 284807 observations, among which $n_{inliers} = 284315$ items are inliers and the remaining $n_{outliers} = 492$ are outliers.

In the case of Credit Card dataset we obtain

$$m + n = 284315/2, \quad m = 2000, \quad n = 2000.$$

Arbitrarily, we choose to set

$$l = 132158, \quad m = 9999, \quad n = 2000$$

in order to have exact control of type I errors at the significance level $\alpha = 0.2$.

Load the data and set the parameters ad described above.

```
set.seed(321)

# Initializing parameters
l = 132158
m = 9999
n = 2000
myalpha = n/(m+1)

dataset = read.csv("G:\\Il mio Drive\\PHD\\Progetto di ricerca\\Conformal Inference Project\\Simulazioni
out_ind = which(dataset$Class==1)
in_ind = which(dataset$Class==0)
```

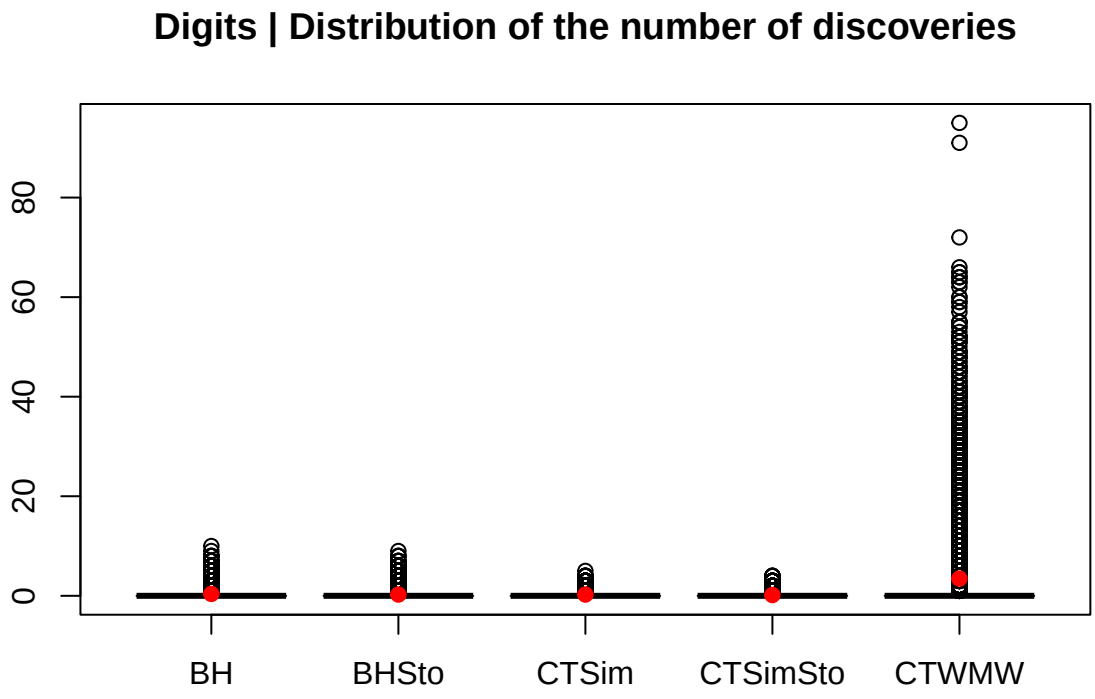
All inliers

We now set the proportion of inliers equal to 1, so that the number of outliers $n_1 = 0$.

```
B=5000

n1=0
n0=n-n1
```

```
tr_ind =  sample(in_ind, size = l)
in_index2 = setdiff(in_ind, tr_ind)
tr = dataset[tr_ind,]
iso.fo = isolation.forest(tr, ndim=ncol(dataset), ntrees=150, sample_size = 256,
                          nthreads=parallel::detectCores(),
                          scoring_metric = "depth", output_score = TRUE)

mycrit = nout::critWMW(m=n,n=m,alpha=myalpha)

d_WMW = rep(0,B)
d_Simes = rep(0,B)
d_StoSimes = rep(0,B)
d_BH = rep(0,B)
d_StoBH = rep(0,B)
uniques = rep(0,B)
#resU = rep(0,B)
N=n+m

for (b in 1:B){
  in_index3 = sample(in_index2, size = N)
  cal_ind = in_index3[1:m]
  te_ind = in_index3[(m+1):N]
  cal = dataset[cal_ind,]
  te = dataset[te_ind,]

  S_cal = predict.isolation_forest(iso.fo$model, cal, type = "score")
  S_te = predict.isolation_forest(iso.fo$model, te, type = "score")
  #U_i = sapply(1:n, function(i) sum(S_te[i]>S_cal))
  #U = sum(U_i)

  d_WMW[b] = nout::d_mannwhitney(S_Y=S_te, S_X=S_cal, crit = mycrit)
  #resU[b] = U >=mycrit$crit.vals[1]
  d_Simes[b] = nout::d_Simes(S_X=S_cal, S_Y=S_te, alpha=myalpha)
  d_StoSimes[b] = nout::d_StoreySimes(S_X=S_cal, S_Y=S_te, alpha=myalpha)$d
  d_BH[b] = nout::d_benjhoch(S_X=S_cal, S_Y=S_te, alpha=myalpha)
  d_StoBH[b] = nout::d_StoreyBH(S_X=S_cal, S_Y=S_te, alpha=myalpha)
  uniques[b] = length(unique(c(S_cal, S_te)))
}


discoveries = as.data.frame(cbind("d_BH"=d_BH,
                                  "d_StoBH"=d_StoBH,
                                  "d_Simes"=d_Simes,
                                  "d_StoSimes"=d_StoSimes,
                                  "d_WMW"=d_WMW))
colnames(discoveries) = c("BH", "BHSto", "CTSim", "CTSimSto", "CTWMW")
mean.discoveries = apply(discoveries, MARGIN = 2, FUN = mean)

powerGlobalNull = as.data.frame(cbind("d_BH"=d_BH>0,
                                      "d_StoBH"=d_StoBH>0,
                                      "d_Simes"=d_Simes>0,
                                      "d_StoSimes"=d_StoSimes>0,
                                      "d_WMW"=d_WMW>0))
```

```
colnames(powerGlobalNull) = c("BH", "BHSto", "CTSim", "CTSimSto", "CTWMW")
mean.powerGlobalNull = apply(powerGlobalNull, MARGIN = 2, FUN = mean)

#prova.d_WMW = mean(d_WMW>0)
#prova.resU = mean(resU)


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




```
mean.discoveries
```

```
##       BH    BHSto    CTSim CTSimSto    CTWMW
##   0.3736   0.2486   0.2510   0.1488   3.4598
```

```
mean.powerGlobalNull
```

```
##       BH    BHSto    CTSim CTSimSto    CTWMW
##   0.2026   0.1200   0.2026   0.1200   0.2052
```

```
#prova.d_WMW
#prova.resU

res=list(#"prova.d_WMW"=prova.d_WMW,
         #"prova.resU"=prova.resU,
         #"resU" = resU,
         "uniques"=uniques,
         "discoveries"=discoveries,
```

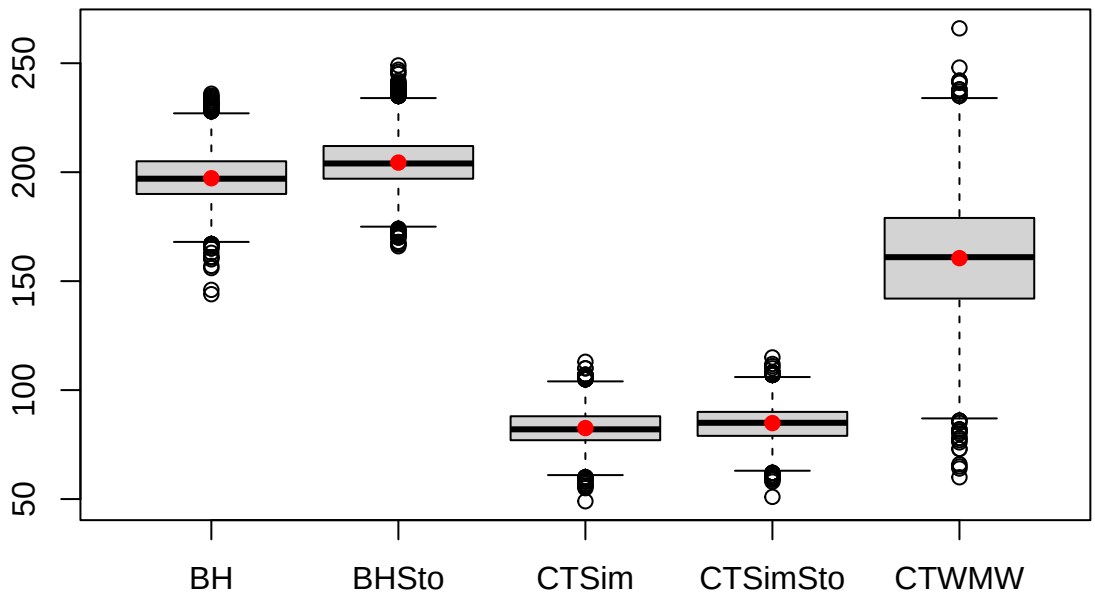
```
          "mean.discoveries" = mean.discoveries,
          "powerGlobalNull"=powerGlobalNull,
          "mean.powerGlobalNull"=mean.powerGlobalNull,
          "n1"=n1,
          "alpha"=myalpha)

resCredit0 = res
save(resCredit0,
     file="~/nout/trials/RealData/PowerStudy/New&TidyNoSimuFunction/alpha0.2/CreditCardOnly0.2/resCredi
```

10% outliers

We now set the proportion of inliers equal to 0.9. Referring to Credit Card dataset we have that the number of inliers is $n_0 = 1800$ and the number of outliers is $n_1 = 200$.

```
B=5000

n1=round(0.1*n)
n0=n-n1

tr_ind = sample(in_ind, size = l)
in_index2 = setdiff(in_ind, tr_ind)
tr = dataset[tr_ind,]
iso.fo = isolation.forest(tr, ndim=ncol(dataset), ntrees=150, sample_size = 256,
                          nthreads=parallel::detectCores(),
                          scoring_metric = "depth", output_score = TRUE)

mycrit = nout::critWMW(m=n,n=m,alpha=myalpha)

d_WMW = rep(0,B)
d_Simes = rep(0,B)
d_StoSimes = rep(0,B)
d_BH = rep(0,B)
d_StoBH = rep(0,B)
#resU = rep(0,B)
uniques = rep(0,B)
N=m+n0

for (b in 1:B){
  in_index3 = sample(in_index2, size = N)
  cal_ind = in_index3[1:m]
  tein_ind = in_index3[(m+1):N]
  teout_ind = sample(out_ind, size = n1)
  cal = dataset[cal_ind,]
  te = dataset[c(tein_ind,teout_ind) ,]

  S_cal = predict.isolation_forest(iso.fo$model, cal, type = "score")
  S_te = predict.isolation_forest(iso.fo$model, te, type = "score")
  #U_i = sapply(1:n, function(i) sum(S_te[i]>S_cal))
  #U = sum(U_i)

  d_WMW[b] = nout::d_mannwhitney(S_Y=S_te, S_X=S_cal, crit = mycrit)
  #resU[b] = U >=mycrit$crit.vals[1]
  d_Simes[b] = nout::d_Simes(S_X=S_cal, S_Y=S_te, alpha=myalpha)
```

```
  d_StoSimes[b] = nout::d_StoreySimes(S_X=S_cal, S_Y=S_te, alpha=myalpha)$d
  d_BH[b] = nout::d_benjhoch(S_X=S_cal, S_Y=S_te, alpha=myalpha)
  d_StoBH[b] = nout::d_StoreyBH(S_X=S_cal, S_Y=S_te, alpha=myalpha)
  uniques[b] = length(unique(c(S_cal, S_te)))
}


discoveries = as.data.frame(cbind("d_BH"=d_BH,
                                  "d_StoBH"=d_StoBH,
                                  "d_Simes"=d_Simes,
                                  "d_StoSimes"=d_StoSimes,
                                  "d_WMW"=d_WMW))
colnames(discoveries) = c("BH", "BHSto", "CTSim", "CTSimSto", "CTWMW")
mean.discoveries = apply(discoveries, MARGIN = 2, FUN = mean)

powerGlobalNull = as.data.frame(cbind("d_BH"=d_BH>0,
                                      "d_StoBH"=d_StoBH>0,
                                      "d_Simes"=d_Simes>0,
                                      "d_StoSimes"=d_StoSimes>0,
                                      "d_WMW"=d_WMW>0))
colnames(powerGlobalNull) = c("BH", "BHSto", "CTSim", "CTSimSto", "CTWMW")
mean.powerGlobalNull = apply(powerGlobalNull, MARGIN = 2, FUN = mean)

#prova.d_WMW = mean(d_WMW>0)
#prova.resU = mean(resU)


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




```
mean.discoveries
```

```
##       BH    BHSto    CTSim CTSimSto    CTWMW
## 197.2170 204.4010  82.5684  84.8530 160.5442
```

```
mean.powerGlobalNull
```

```
##       BH    BHSto    CTSim CTSimSto    CTWMW
##        1        1        1        1        1
```

```
#prova.d_WMW
#prova.resU

res=list(#"prova.d_WMW"=prova.d_WMW,
         #"prova.resU"=prova.resU,
         #"resU" = resU,
         "uniques"=uniques,
         "discoveries"=discoveries,
         "mean.discoveries" = mean.discoveries,
         "powerGlobalNull"=powerGlobalNull,
         "mean.powerGlobalNull"=mean.powerGlobalNull,
         "n1"=n1,
         "alpha"=myalpha)

resCredit10 = res
save(resCredit10,
     file="~/nout/trials/RealData/PowerStudy/New&TidyNoSimuFunction/alpha0.2/CreditCardOnly0.2/resCredi
```