```r
library("scmamp")
```

```r
resultados <- read.csv("test_res_ensemble.csv", header = TRUE, sep = '\t')
tablatst <- cbind(resultados[,1:dim(resultados)[2]])
colnames(tablatst) <- names(resultados)[1:dim(resultados)[2]]
```

```r
colnames(tablatst)
```

```
## [1] "SMOTE" "ROS"   "VAE"   "GAN"
```

```r
tablatst
```

```
##     SMOTE    ROS    VAE    GAN
## 1  0.9105 0.8312 0.8312 0.9182
## 2  0.9821 0.9806 0.9796 0.9800
## 3  0.8942 0.8930 0.8856 0.8853
## 4  1.0000 1.0000 1.0000 0.9976
## 5  0.7653 0.7377 0.7996 0.7718
## 6  0.9330 0.9339 0.9382 0.9313
## 7  0.9331 0.9331 0.9331 0.8729
## 8  0.9607 0.9969 0.9413 0.9782
## 9  0.9589 0.9461 0.9531 0.9537
## 10 0.8574 0.8807 0.8958 0.8723
## 11 0.6142 0.5280 0.6023 0.6535
## 12 0.7295 0.7085 0.6603 0.7039
## 13 0.8944 0.7746 0.8944 0.8944
## 14 0.9965 0.9856 0.9936 0.9821
## 15 0.6108 0.6280 0.6215 0.6416
## 16 0.8944 0.8944 0.7746 0.8944
## 17 0.6639 0.6396 0.7025 0.6722
## 18 0.8511 0.8010 0.7719 0.7800
## 19 1.0000 0.9985 1.0000 1.0000
## 20 0.7916 0.7848 0.7659 0.7401
## 21 0.9879 0.9857 0.9900 0.9945
## 22 0.7849 0.7990 0.7920 0.8369
## 23 0.8986 0.9459 0.9420 0.9383
## 24 0.7409 0.7508 0.7391 0.7299
## 25 0.9401 0.9373 0.9589 0.9400
## 26 0.9650 0.9645 0.9659 0.9595
## 27 0.7226 0.5470 0.7455 0.8022
```

```r
#Test de Fligner-Killeen para la homocedasticidad (no paramétrico, mediana, no normalidad)
fligner.test(x = list(tablatst$SMOTE, tablatst$ROS, tablatst$VAE, tablatst$GAN))
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  list(tablatst$SMOTE, tablatst$ROS, tablatst$VAE, tablatst$GAN)
## Fligner-Killeen:med chi-squared = 1.3056, df = 3, p-value = 0.7278
```

```r
#Aplicación del test de Friedman Aligned Ranks
test_friedman_aligned_ranks <- friedmanAlignedRanksTest(as.matrix(tablatst))
test_friedman_aligned_ranks
```

```
##
##  Friedman's Aligned Rank Test for Multiple Comparisons
##
```

```
## data:  as.matrix(tablatst)
## T = 1.9425, df = 3, p-value = 0.5844
```

```
#Aplicación del test de Friedman Aligned Ranks post-hoc
test_friedman_aligned_ranks_post <- friedmanAlignedRanksPost(as.matrix(tablatst))
test_friedman_aligned_ranks_post
```

```
##           SMOTE       ROS       VAE       GAN
## SMOTE        NA 0.1497833 0.5202055 0.7926587
## ROS   0.1497833        NA 0.4252950 0.2390209
## VAE   0.5202055 0.4252950        NA 0.7038200
## GAN   0.7926587 0.2390209 0.7038200        NA
```

```
#Bergmann and Hommel dynamic correction of p-values
adjustBergmannHommel (test_friedman_aligned_ranks_post)
```

```
##           SMOTE       ROS       VAE       GAN
## SMOTE        NA 0.8986998 1.0000000 1.0000000
## ROS   0.8986998        NA 0.8986998 0.8986998
## VAE   1.0000000 0.8986998        NA 1.0000000
## GAN   1.0000000 0.8986998 1.0000000        NA
```

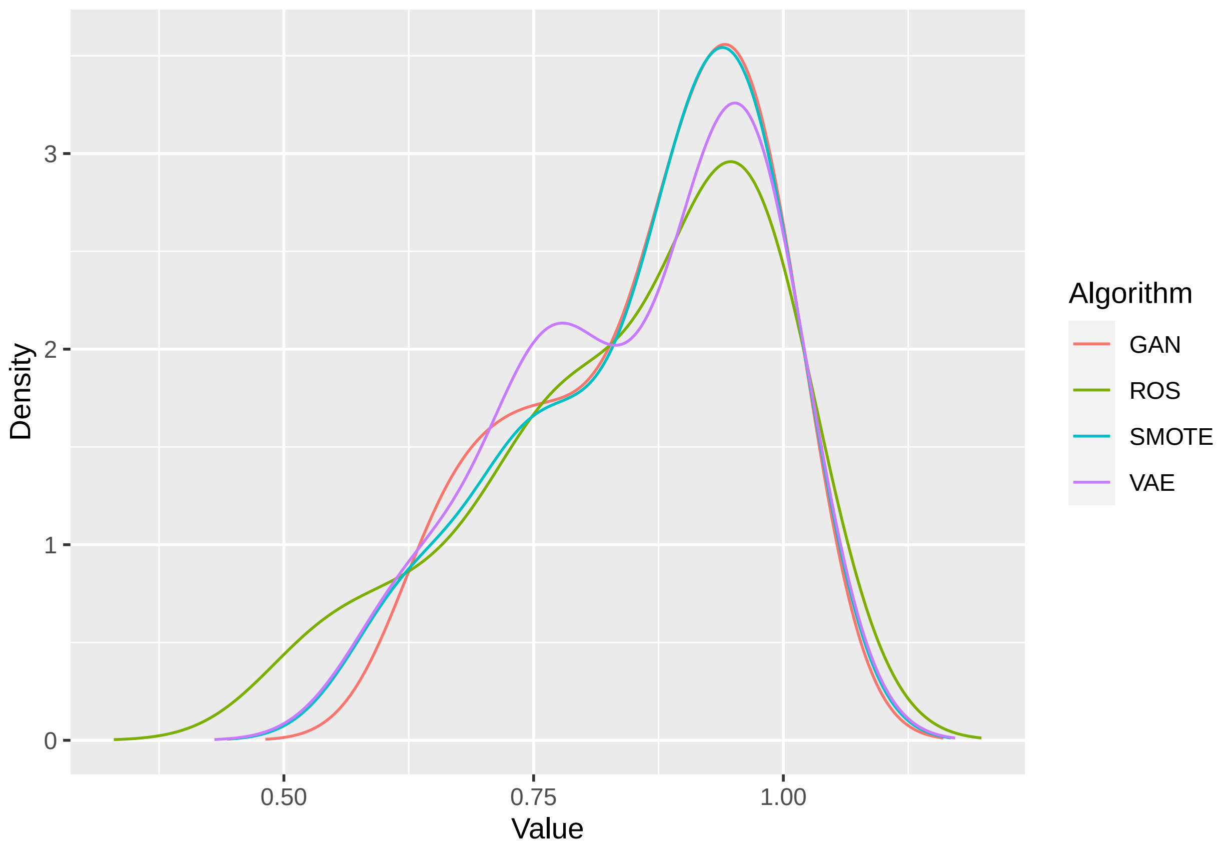
```
quadeTest(as.matrix(tablatst))
```

```
##
##  Quade for Multiple Comparisons
##
## data:  as.matrix(tablatst)
## T = 0.65056, df = 3, p-value = 0.585
```

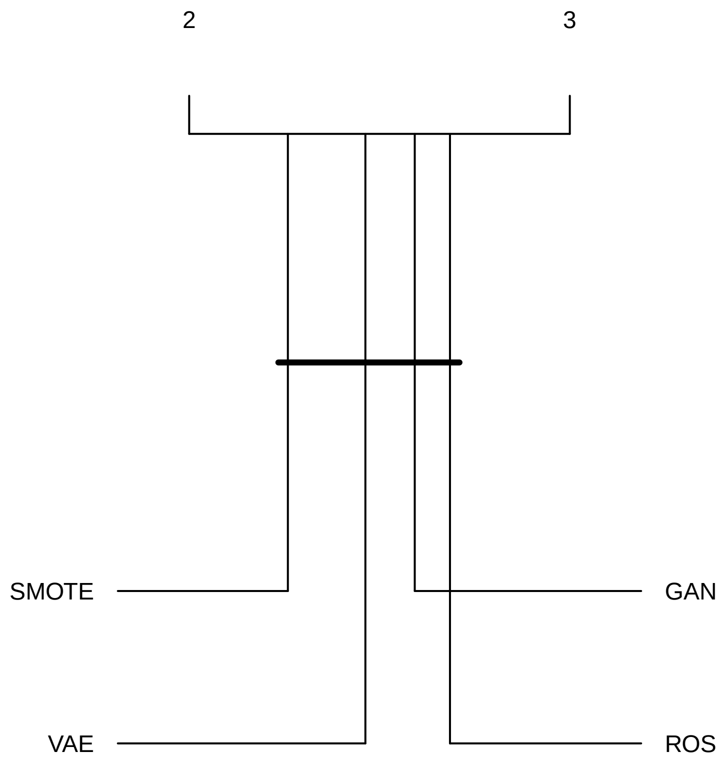
```
quadePost(as.matrix(tablatst))
```

```
##           SMOTE       ROS       VAE       GAN
## SMOTE        NA 0.3540842 0.5455995 0.9785708
## ROS   0.3540842        NA 0.7472033 0.3403078
## VAE   0.5455995 0.7472033        NA 0.5278915
## GAN   0.9785708 0.3403078 0.5278915        NA
```

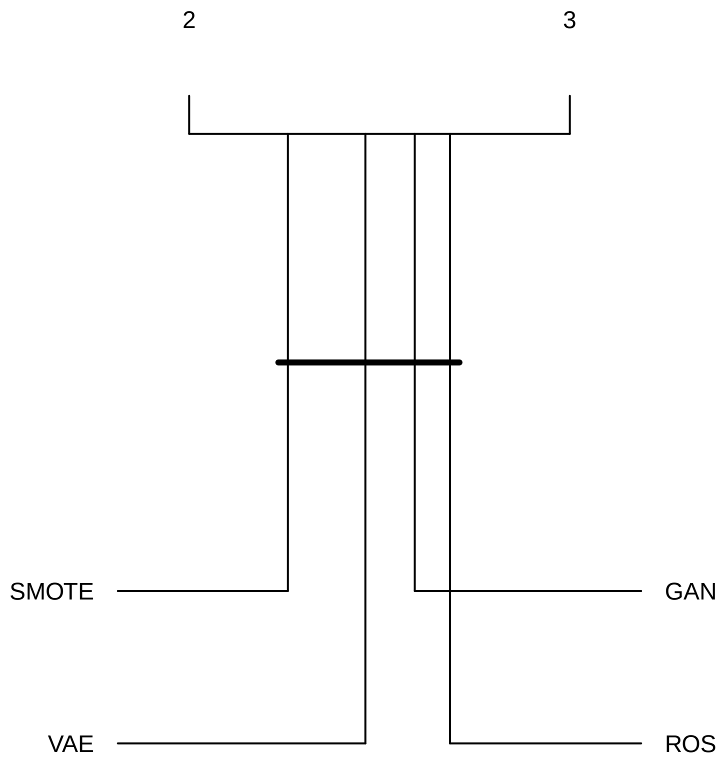
```
plotDensities(as.matrix(tablatst))
```



```
test <- postHocTest(as.matrix(tablatst), test="aligned ranks", correct="bergmann", use.rank=TRUE)
plotRanking(pvalues=test$corrected.pval, summary=test$summary, alpha=0.05)
```



```
test <- postHocTest(as.matrix(tablatst), test="quade", correct="bergmann", use.rank=TRUE)
plotRanking(pvalues=test$corrected.pval, summary=test$summary, alpha=0.05)
```



```
##TABLA NORMALIZADA - smote (other) vs vae (ref) para WILCOXON
# + 0.1 porque wilcox R falla para valores == 0 en la tabla

difs <- (tablatst[,1] - tablatst[,3]) / tablatst[,1]
wilc_1_2 <- cbind(ifelse (difs<0, abs(difs)+0.1, 0+0.1), ifelse (difs>0,    abs(difs)+0.1, 0+0.1))
colnames(wilc_1_2) <- c(colnames(tablatst)[1], colnames(tablatst)[3])
head(wilc_1_2)
```

```
##          SMOTE       VAE
## [1,] 0.1000000 0.1870950
## [2,] 0.1000000 0.1025456
## [3,] 0.1000000 0.1096175
## [4,] 0.1000000 0.1000000
## [5,] 0.1448190 0.1000000
## [6,] 0.1055734 0.1000000
```

```
#Aplicación del test de WILCOXON
SMvsVAEtst <- wilcox.test(wilc_1_2[,1], wilc_1_2[,2], alternative = "two.sided", paired=TRUE)
```

```
## Warning in wilcox.test.default(wilc_1_2[, 1], wilc_1_2[, 2], alternative =
## "two.sided", : cannot compute exact p-value with zeroes
```

```
Rmas <- SMvsVAEtst$statistic
pvalue <- SMvsVAEtst$p.value
SMvsVAEtst <- wilcox.test(wilc_1_2[,2], wilc_1_2[,1], alternative = "two.sided", paired=TRUE)
```

```
## Warning in wilcox.test.default(wilc_1_2[, 2], wilc_1_2[, 1], alternative =
## "two.sided", : cannot compute exact p-value with zeroes
```

```
Rmenos <- SMvsVAEtst$statistic
Rmas
```

```
##   V
## 123
```

```
Rmenos
```

```
##   V
## 153
```

```
pvalue
```

```
## [1] 0.6592008
```

```
##TABLA NORMALIZADA - smote (other) vs gan (ref) para WILCOXON
# + 0.1 porque wilcox R falla para valores == 0 en la tabla

difs <- (tablatst[,1] - tablatst[,4]) / tablatst[,1]
wilc_1_2 <- cbind(ifelse (difs<0, abs(difs)+0.1, 0+0.1), ifelse (difs>0,    abs(difs)+0.1, 0+0.1))
colnames(wilc_1_2) <- c(colnames(tablatst)[1], colnames(tablatst)[4])
head(wilc_1_2)
```

```
##          SMOTE       GAN
## [1,] 0.1084569 0.1000000
## [2,] 0.1000000 0.1021383
## [3,] 0.1000000 0.1099530
## [4,] 0.1000000 0.1024000
## [5,] 0.1084934 0.1000000
## [6,] 0.1000000 0.1018221
```

```
#Aplicación del test de WILCOXON
SMvsGANtst <- wilcox.test(wilc_1_2[,1], wilc_1_2[,2], alternative = "two.sided", paired=TRUE)
```

```
## Warning in wilcox.test.default(wilc_1_2[, 1], wilc_1_2[, 2], alternative =
## "two.sided", : cannot compute exact p-value with zeroes
```

```
Rmas <- SMvsGANtst$statistic
pvalue <- SMvsGANtst$p.value
SMvsGANtst <- wilcox.test(wilc_1_2[,2], wilc_1_2[,1], alternative = "two.sided", paired=TRUE)
```

```
## Warning in wilcox.test.default(wilc_1_2[, 2], wilc_1_2[, 1], alternative =
## "two.sided", : cannot compute exact p-value with zeroes
```

```
Rmenos <- SMvsGANtst$statistic
Rmas
```

```
##   V
## 164
```

```
Rmenos
```

```
##   V
## 136
```

```
pvalue
```

```
## [1] 0.6997083
```

```
##TABLA NORMALIZADA - vae (other) vs gan (ref) para WILCOXON
# + 0.1 porque wilcox R falla para valores == 0 en la tabla
```

```r
difs <- (tablatst[,3] - tablatst[,4]) / tablatst[,3]
wilc_1_2 <- cbind(ifelse (difs<0, abs(difs)+0.1, 0+0.1), ifelse (difs>0,    abs(difs)+0.1, 0+0.1))
colnames(wilc_1_2) <- c(colnames(tablatst)[3], colnames(tablatst)[4])
head(wilc_1_2)
```

```
##            VAE       GAN
## [1,] 0.2046679 0.1000000
## [2,] 0.1004083 0.1000000
## [3,] 0.1000000 0.1003388
## [4,] 0.1000000 0.1024000
## [5,] 0.1000000 0.1347674
## [6,] 0.1000000 0.1073545
```

```r
#Aplicación del test de WILCOXON
VAEvsGANtst <- wilcox.test(wilc_1_2[,1], wilc_1_2[,2], alternative = "two.sided", paired=TRUE)
```

```
## Warning in wilcox.test.default(wilc_1_2[, 1], wilc_1_2[, 2], alternative =
## "two.sided", : cannot compute exact p-value with zeroes
```

```r
Rmas <- VAEvsGANtst$statistic
pvalue <- VAEvsGANtst$p.value
VAEvsGANtst <- wilcox.test(wilc_1_2[,2], wilc_1_2[,1], alternative = "two.sided", paired=TRUE)
```

```
## Warning in wilcox.test.default(wilc_1_2[, 2], wilc_1_2[, 1], alternative =
## "two.sided", : cannot compute exact p-value with zeroes
```

```r
Rmenos <- VAEvsGANtst$statistic
Rmas
```

```
##   V
## 185
```

```r
Rmenos
```

```
##   V
## 140
```

```r
pvalue
```

```
## [1] 0.5538827
```