R Notebook

Parametros:

Measure = Matthews correlation coefficient
Columns = learner
Performance = tuning_measure
Filter keys = sampling, weight_space, ruspool
Filter values = FALSE, FALSE, FALSE

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

Tratamento dos dados

Carregando data set compilado

```r
ds = read.csv("/home/rodrigo/Dropbox/UNICAMP/IC/estudo_cost_learning/SummaryResults/summary_compilation
summary(ds)
```

```
##                 learner       weight_space
##  classif.ksvm        :17100   Mode :logical
##  classif.randomForest:17100   FALSE:41040
##  classif.xgboost     :17100   TRUE :10260
##                               NA's :0
##
##
##
##                                   measure        sampling      ruspool
##  Accuracy                       :10260   ADASYN:10260   Mode :logical
##  Area under the curve           :10260   FALSE :30780   FALSE:41040
##  F1 measure                     :10260   SMOTE :10260   TRUE :10260
##  G-mean                         :10260                  NA's :0
##  Matthews correlation coefficient:10260
##
##
##  tuning_measure     holdout_measure    holdout_measure_residual
##  Min.   :-0.1277    Min.   :-0.2120    Min.   :-0.4658
##  1st Qu.: 0.5924    1st Qu.: 0.3114    1st Qu.: 0.1648
##  Median : 0.9624    Median : 0.8193    Median : 0.5192
##  Mean   : 0.7570    Mean   : 0.6469    Mean   : 0.5099
##  3rd Qu.: 0.9965    3rd Qu.: 0.9879    3rd Qu.: 0.8636
##  Max.   : 1.0000    Max.   : 1.0000    Max.   : 1.0000
##  NA's   :1761       NA's   :1761       NA's   :1761
##  iteration_count                      dataset         imba.rate
##  Min.   :1       abalone                  : 900   Min.   :0.0010
##  1st Qu.:1       adult                    : 900   1st Qu.:0.0100
##  Median :2       bank                     : 900   Median :0.0300
##  Mean   :2       car                      : 900   Mean   :0.0286
##  3rd Qu.:3       cardiotocography-10clases: 900   3rd Qu.:0.0500
##  Max.   :3       cardiotocography-3clases : 900   Max.   :0.0500
```

```
##  NA's   :1761   (Other)                    :45900
```

Filtrando pela metrica

```
ds = filter(ds, measure == params$measure)
```

Filtrando o data set

```
if(params$filter_keys != 'NULL' && !is.null(params$filter_keys)){
  ds = filter_at(ds, .vars = params$filter_keys, .vars_predicate = any_vars(. == params$filter_values))
}

summary(ds)
```

```
##                 learner       weight_space
##   classif.ksvm         :3420   Mode :logical
##   classif.randomForest:3420   FALSE:8208
##   classif.xgboost      :3420   TRUE :2052
##                                NA's :0
##
##
##
##                                      measure          sampling      ruspool
##   Accuracy                        :   0   ADASYN:2052   Mode :logical
##   Area under the curve            :   0   FALSE :6156   FALSE:8208
##   F1 measure                      :   0   SMOTE :2052   TRUE :2052
##   G-mean                          :   0                 NA's :0
##   Matthews correlation coefficient:10260
##
##
##   tuning_measure    holdout_measure    holdout_measure_residual
##   Min.    :-0.1277   Min.    :-0.2120   Min.    :-0.4658
##   1st Qu.: 0.2088   1st Qu.: 0.0000   1st Qu.: 0.0067
##   Median : 0.7306   Median : 0.4258   Median : 0.1812
##   Mean    : 0.6065   Mean    : 0.4333   Mean    : 0.2883
##   3rd Qu.: 0.9851   3rd Qu.: 0.8098   3rd Qu.: 0.5031
##   Max.    : 1.0000   Max.    : 1.0000   Max.    : 1.0000
##   NA's    :357       NA's    :357       NA's    :357
##   iteration_count                       dataset       imba.rate
##   Min.   :1       abalone                   : 180   Min.    :0.0010
##   1st Qu.:1       adult                     : 180   1st Qu.:0.0100
##   Median :2       bank                      : 180   Median :0.0300
##   Mean   :2       car                       : 180   Mean    :0.0286
##   3rd Qu.:3       cardiotocography-10clases: 180   3rd Qu.:0.0500
##   Max.    :3       cardiotocography-3clases : 180   Max.    :0.0500
##   NA's    :357    (Other)                   :9180
```

Computando as médias das iteracoes

```
ds = group_by(ds, learner , weight_space , measure , sampling , ruspool , dataset , imba.rate)
ds = summarise(ds, tuning_measure = mean(tuning_measure), holdout_measure = mean(holdout_measure),
            holdout_measure_residual = mean(holdout_measure_residual))

ds = as.data.frame(ds)
```

Criando dataframe

```r
# Dividindo o ds em n, um para cada técnica
splited_df = ds %>% group_by_at(.vars = params$columns) %>% do(vals = as.data.frame(.)) %>% select(vals

# Juntando cada uma das partes horizontalmente em um data set
df_tec_wide = do.call("cbind", splited_df)

# Renomeando duplicacao de nomes
colnames(df_tec_wide) = make.unique(colnames(df_tec_wide))

# Selecionando apenas as medidas da performance escolhida
df_tec_wide_residual = select(df_tec_wide, matches(paste("^", params$performance, "$|", params$performan

# Renomeando colunas
new_names = NULL
for(i in (1:length(splited_df))){
  id = toString(sapply(splited_df[[i]][1, params$columns], as.character))
  new_names = c(new_names, id)
}
colnames(df_tec_wide_residual) = new_names

# Verificando a dimensao do df
dim(df_tec_wide_residual)
```

```
## [1] 1140    3
```

```r
# Renomeando a variavel
df = df_tec_wide_residual

summary(df)
```

```
##   classif.ksvm      classif.randomForest classif.xgboost
##  Min.   :-0.003311   Min.   :-0.00202    Min.   :-0.004855
##  1st Qu.: 0.177979   1st Qu.: 0.40585    1st Qu.: 0.142875
##  Median : 0.687480   Median : 0.85295    Median : 0.630475
##  Mean   : 0.589182   Mean   : 0.67790    Mean   : 0.558642
##  3rd Qu.: 0.978018   3rd Qu.: 0.99088    3rd Qu.: 0.974107
##  Max.   : 1.000000   Max.   : 1.00000    Max.   : 1.000000
##  NA's   :10          NA's   :106         NA's   :3
```
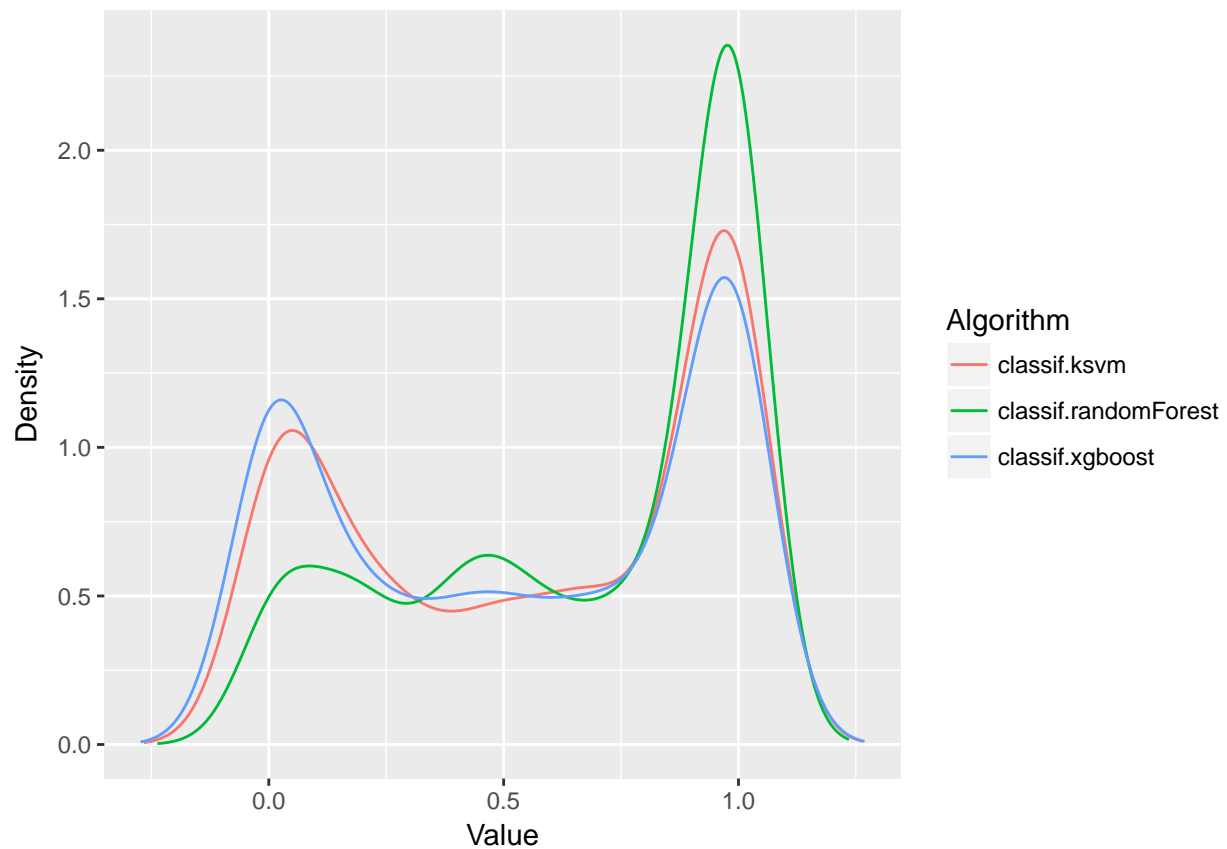
Fazendo teste de normalidade

```r
plotDensities(data = na.omit(df))
```

Testando as diferencas

```
friedmanTest(df)
```

```
## 
##  Friedman's rank sum test
## 
## data:  df
## Friedman's chi-squared = 78.69, df = 2, p-value < 2.2e-16
```

Testando as diferencas par a par

```
test <- nemenyiTest (df, alpha=0.05)
abs(test$diff.matrix) > test$statistic
```

```
##       classif.ksvm classif.randomForest classif.xgboost
## [1,]         FALSE                 TRUE            TRUE
## [2,]          TRUE                FALSE            TRUE
## [3,]          TRUE                 TRUE           FALSE
```

Plotando grafico de Critical Diference

```
result = tryCatch({
    plotCD(df, alpha=0.05, cex = 0.35)
}, error = function(e) {})
```

CD

1   2   3

classif.randomForest

classif.xgboost

classif.ksvm