R Notebook

Parametros:

Measure = Accuracy
Columns = sampling, weight_space, underbagging, learner
Performance = holdout_measure
Filter keys = imba.rate
Filter values = 0.05

```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_

ds = filter(ds, learner != "classif.rusboost")
summary(ds)
```

```
##                  learner      weight_space
##  classif.ksvm        :17100   Mode :logical
##  classif.randomForest:17100   FALSE:41040
##  classif.rusboost    :    0   TRUE :10260
##  classif.xgboost     :17100   NA's :0
##
##
##
##                             measure        sampling     underbagging
##  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.: 0.6911   1st Qu.: 0.4001   1st Qu.: 0.1994
##  Median : 0.9700   Median : 0.8571   Median : 0.5581
##  Mean   : 0.7903   Mean   : 0.6718   Mean   : 0.5298
##  3rd Qu.: 0.9975   3rd Qu.: 0.9900   3rd Qu.: 0.8755
##  Max.   : 1.0000   Max.   : 1.0000   Max.   : 1.0000
##  NA's   :1077      NA's   :1077      NA's   :1077
##  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   :1077    (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)){
  dots = paste0(params$filter_keys," == '",params$filter_values,"'")
  ds = filter_(ds, .dots = dots)
}

summary(ds)
```

```
##                    learner     weight_space
##  classif.ksvm        :1230   Mode :logical
##  classif.randomForest:1230   FALSE:2952
##  classif.rusboost    :   0   TRUE :738
##  classif.xgboost     :1230   NA's :0
##
##
##
##                              measure       sampling    underbagging
##  Accuracy                        :3690   ADASYN: 738   Mode :logical
##  Area under the curve            :   0   FALSE :2214   FALSE:2952
##  F1 measure                      :   0   SMOTE : 738   TRUE :738
##  G-mean                          :   0                 NA's :0
##  Matthews correlation coefficient:   0
##
##
##  tuning_measure   holdout_measure   holdout_measure_residual
##  Min.   :0.2470   Min.   :0.04739   Min.   :0.0367
##  1st Qu.:0.9494   1st Qu.:0.94505   1st Qu.:0.3902
##  Median :0.9688   Median :0.96078   Median :0.7223
##  Mean   :0.9425   Mean   :0.93413   Mean   :0.6602
##  3rd Qu.:0.9908   3rd Qu.:0.98413   3rd Qu.:0.9315
##  Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
##  NA's   :42       NA's   :42        NA's   :42
##  iteration_count          dataset       imba.rate
##  Min.   :1       abalone      :  45   Min.   :0.05
##  1st Qu.:1       adult        :  45   1st Qu.:0.05
##  Median :2       annealing    :  45   Median :0.05
##  Mean   :2       arrhythmia   :  45   Mean   :0.05
##  3rd Qu.:3       balance-scale:  45   3rd Qu.:0.05
##  Max.   :3       bank         :  45   Max.   :0.05
##  NA's   :42      (Other)      :3420
```

Computando as médias das iteracoes

```
ds = group_by(ds, learner , weight_space , measure , sampling , underbagging , 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

```
# 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$performa

# 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] 82 15
```

```
# Renomeando a variavel
df = df_tec_wide_residual

head(df)
```

```
##   ADASYN, FALSE, FALSE, classif.ksvm
## 1                          0.8872222
## 2                          0.9183164
## 3                          0.9569161
## 4                          0.9607843
## 5                          1.0000000
## 6                          0.9424603
##   ADASYN, FALSE, FALSE, classif.randomForest
## 1                                  0.9050000
## 2                                         NA
## 3                                  0.9773243
## 4                                  0.9673203
## 5                                  1.0000000
## 6                                  0.9380952
##   ADASYN, FALSE, FALSE, classif.xgboost FALSE, FALSE, FALSE, classif.ksvm
## 1                             0.9177778                          0.9227778
## 2                             0.9538087                          0.9418284
## 3                             0.9818594                          0.9478458
## 4                             0.9738562                          0.9607843
## 5                             1.0000000                          1.0000000
## 6                             0.9428571                          0.9500000
##   FALSE, FALSE, FALSE, classif.randomForest
## 1                                 0.9500000
## 2                                 0.9633545
## 3                                 0.9750567
## 4                                 0.9673203
## 5                                 1.0000000
## 6                                 0.9500000
##   FALSE, FALSE, FALSE, classif.xgboost FALSE, FALSE, TRUE, classif.ksvm
## 1                            0.9500000                        0.5938889
## 2                            0.9649561                        0.8868601
## 3                            0.9637188                        0.9319728
## 4                            0.9803922                        0.8758170
## 5                            1.0000000                        0.9900498
## 6                            0.9440476                        0.5579365
##   FALSE, FALSE, TRUE, classif.randomForest
## 1                                0.5933333
## 2                                       NA
## 3                                0.8775510
## 4                                0.9084967
## 5                                0.9900498
## 6                                0.8071429
##   FALSE, FALSE, TRUE, classif.xgboost FALSE, TRUE, FALSE, classif.ksvm
## 1                           0.6255556                        0.9338889
## 2                           0.8145941                        0.9464412
## 3                           0.8639456                        0.9455782
## 4                           0.9215686                        0.9607843
## 5                           0.9452736                        1.0000000
## 6                           0.8035714                        0.9500000
##   FALSE, TRUE, FALSE, classif.randomForest
## 1                                0.9500000
## 2                                0.9636107
## 3                                0.9705215
## 4                                0.9803922
## 5                                1.0000000
## 6                                0.9500000
##   FALSE, TRUE, FALSE, classif.xgboost SMOTE, FALSE, FALSE, classif.ksvm
## 1                           0.9500000                         0.8933333
## 2                           0.9627138                         0.9201102
## 3                           0.9659864                         0.9546485
## 4                           0.9803922                         0.9607843
## 5                           1.0000000                         0.9850746
## 6                           0.9496032                         0.9420635
##   SMOTE, FALSE, FALSE, classif.randomForest
## 1                                 0.9088889
## 2                                 0.9406112
## 3                                 0.9818594
## 4                                 0.9673203
## 5                                 1.0000000
## 6                                 0.9341270
##   SMOTE, FALSE, FALSE, classif.xgboost
## 1                            0.9188889
## 2                            0.9566917
## 3                            0.9841270
## 4                            0.9869281
## 5                            1.0000000
## 6                            0.9400794
```

```r
summary(df)
```

```
##  ADASYN, FALSE, FALSE, classif.ksvm
##  Min.   :0.5434
##  1st Qu.:0.9457
##  Median :0.9569
##  Mean   :0.9460
##  3rd Qu.:0.9748
##  Max.   :1.0000
##  NA's   :1
##  ADASYN, FALSE, FALSE, classif.randomForest
##  Min.   :0.5030
##  1st Qu.:0.9488
##  Median :0.9726
##  Mean   :0.9521
##  3rd Qu.:0.9899
##  Max.   :1.0000
##  NA's   :4
##  ADASYN, FALSE, FALSE, classif.xgboost FALSE, FALSE, FALSE, classif.ksvm
##  Min.   :0.5212                        Min.   :0.9228
##  1st Qu.:0.9372                        1st Qu.:0.9501
##  Median :0.9741                        Median :0.9570
##  Mean   :0.9493                        Mean   :0.9614
##  3rd Qu.:0.9875                        3rd Qu.:0.9737
##  Max.   :1.0000                        Max.   :1.0000
##
##  FALSE, FALSE, FALSE, classif.randomForest
##  Min.   :0.9333
##  1st Qu.:0.9554
##  Median :0.9733
##  Mean   :0.9731
##  3rd Qu.:0.9894
##  Max.   :1.0000
##  NA's   :1
##  FALSE, FALSE, FALSE, classif.xgboost FALSE, FALSE, TRUE, classif.ksvm
##  Min.   :0.9333                       Min.   :0.07425
##  1st Qu.:0.9576                       1st Qu.:0.67863
##  Median :0.9752                       Median :0.90664
##  Mean   :0.9728                       Mean   :0.80542
##  3rd Qu.:0.9863                       3rd Qu.:0.96825
##  Max.   :1.0000                       Max.   :0.99797
##
##  FALSE, FALSE, TRUE, classif.randomForest
##  Min.   :0.3968
##  1st Qu.:0.7836
##  Median :0.8970
##  Mean   :0.8497
##  3rd Qu.:0.9662
##  Max.   :1.0000
##  NA's   :3
##  FALSE, FALSE, TRUE, classif.xgboost FALSE, TRUE, FALSE, classif.ksvm
##  Min.   :0.3636                      Min.   :0.9333
##  1st Qu.:0.7700                      1st Qu.:0.9501
##  Median :0.8713                      Median :0.9558
##  Mean   :0.8438                      Mean   :0.9615                 
##  3rd Qu.:0.9519                      3rd Qu.:0.9731                 
##  Max.   :1.0000                      Max.   :1.0000                 
##                                                                     
##  FALSE, TRUE, FALSE, classif.randomForest
##  Min.   :0.9333                      
##  1st Qu.:0.9583                      
##  Median :0.9709                      
##  Mean   :0.9729                      
##  3rd Qu.:0.9894                      
##  Max.   :1.0000                      
##  NA's   :1                           
##  FALSE, TRUE, FALSE, classif.xgboost SMOTE, FALSE, FALSE, classif.ksvm
##  Min.   :0.9380                      Min.   :0.5713                 
##  1st Qu.:0.9547                      1st Qu.:0.9432                 
##  Median :0.9735                      Median :0.9552                 
##  Mean   :0.9724                      Mean   :0.9461                 
##  3rd Qu.:0.9878                      3rd Qu.:0.9706                 
##  Max.   :1.0000                      Max.   :1.0000                 
##                                                                     
##  SMOTE, FALSE, FALSE, classif.randomForest
##  Min.   :0.5333                           
##  1st Qu.:0.9383                           
##  Median :0.9724                           
##  Mean   :0.9524                           
##  3rd Qu.:0.9917                           
##  Max.   :1.0000                           
##  NA's   :4                                
##  SMOTE, FALSE, FALSE, classif.xgboost
##  Min.   :0.5152                      
##  1st Qu.:0.9380                      
##  Median :0.9740                      
##  Mean   :0.9527                      
##  3rd Qu.:0.9909                      
##  Max.   :1.0000                      
##
```

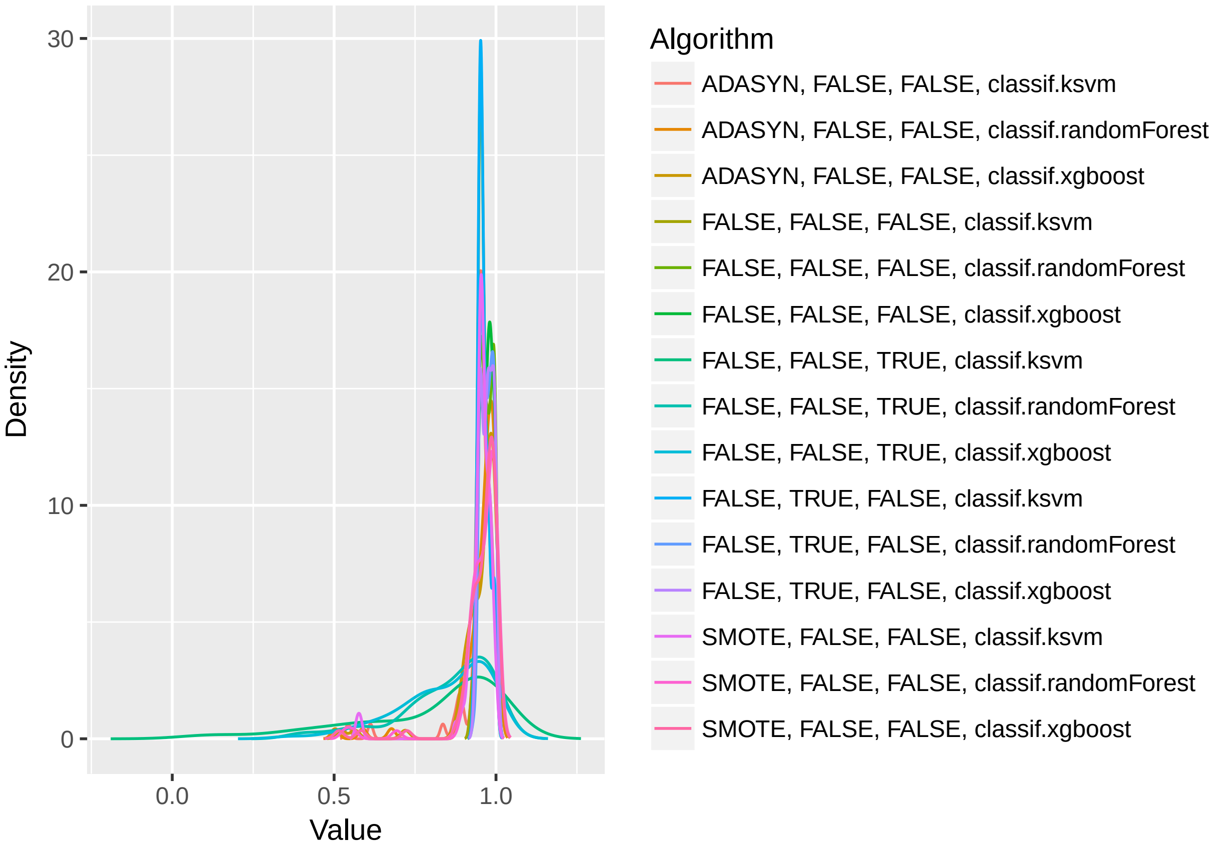
Verificando a média de cada coluna selecionada

```
for(i in (1:dim(df)[2])){
  print(paste("Media da coluna ", colnames(df)[i], " = ", mean(df[,i], na.rm = TRUE), sep=""))
}
```

```
## [1] "Media da coluna ADASYN, FALSE, FALSE, classif.ksvm = 0.946020162406282"
## [1] "Media da coluna ADASYN, FALSE, FALSE, classif.randomForest = 0.952094731045437"
## [1] "Media da coluna ADASYN, FALSE, FALSE, classif.xgboost = 0.949302471873812"
## [1] "Media da coluna FALSE, FALSE, FALSE, classif.ksvm = 0.961419018550742"
## [1] "Media da coluna FALSE, FALSE, FALSE, classif.randomForest = 0.973087334234863"
## [1] "Media da coluna FALSE, FALSE, FALSE, classif.xgboost = 0.972820661306621"
## [1] "Media da coluna FALSE, FALSE, TRUE, classif.ksvm = 0.805419158104711"
## [1] "Media da coluna FALSE, FALSE, TRUE, classif.randomForest = 0.849703118198864"
## [1] "Media da coluna FALSE, FALSE, TRUE, classif.xgboost = 0.843772344908859"
## [1] "Media da coluna FALSE, TRUE, FALSE, classif.ksvm = 0.961501896195199"
## [1] "Media da coluna FALSE, TRUE, FALSE, classif.randomForest = 0.972942514880323"
## [1] "Media da coluna FALSE, TRUE, FALSE, classif.xgboost = 0.972416356558692"
## [1] "Media da coluna SMOTE, FALSE, FALSE, classif.ksvm = 0.94614439194753"
## [1] "Media da coluna SMOTE, FALSE, FALSE, classif.randomForest = 0.952350207201139"
## [1] "Media da coluna SMOTE, FALSE, FALSE, classif.xgboost = 0.952709335640466"
```

Fazendo teste de normalidade

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



Testando as diferencas

```
friedmanTest(df)
```

```
##
##  Friedman's rank sum test
##
## data:  df
## Friedman's chi-squared = 481.24, df = 14, p-value < 2.2e-16
```

Testando as diferencas par a par

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

```
##       ADASYN, FALSE, FALSE, classif.ksvm
##  [1,]                              FALSE
##  [2,]                              FALSE
##  [3,]                              FALSE
##  [4,]                              FALSE
##  [5,]                               TRUE
##  [6,]                               TRUE
##  [7,]                               TRUE
##  [8,]                               TRUE
##  [9,]                               TRUE
## [10,]                              FALSE
## [11,]                               TRUE
## [12,]                               TRUE
## [13,]                              FALSE
## [14,]                               TRUE
## [15,]                               TRUE
##       ADASYN, FALSE, FALSE, classif.randomForest
##  [1,]                                      FALSE
##  [2,]                                      FALSE
##  [3,]                                      FALSE
##  [4,]                                      FALSE
##  [5,]                                      FALSE
##  [6,]                                       TRUE
##  [7,]                                       TRUE
##  [8,]                                       TRUE
##  [9,]                                       TRUE
## [10,]                                      FALSE
## [11,]                                      FALSE
## [12,]                                      FALSE
## [13,]                                      FALSE
## [14,]                                      FALSE
## [15,]                                      FALSE
##       ADASYN, FALSE, FALSE, classif.xgboost
##  [1,]                                 FALSE
##  [2,]                                 FALSE
##  [3,]                                 FALSE
##  [4,]                                 FALSE
##  [5,]                                 FALSE
##  [6,]                                  TRUE
##  [7,]                                  TRUE
##  [8,]                                  TRUE
##  [9,]                                  TRUE
## [10,]                                 FALSE
## [11,]                                 FALSE
## [12,]                                 FALSE
## [13,]                                  TRUE
## [14,]                                 FALSE
## [15,]                                 FALSE
##       FALSE, FALSE, FALSE, classif.ksvm
##  [1,]                             FALSE
##  [2,]                             FALSE
##  [3,]                             FALSE
##  [4,]                             FALSE
##  [5,]                              TRUE
##  [6,]                              TRUE
##  [7,]                              TRUE
##  [8,]                              TRUE
##  [9,]                              TRUE
## [10,]                             FALSE
## [11,]                              TRUE
## [12,]                              TRUE
## [13,]                             FALSE
## [14,]                             FALSE
## [15,]                             FALSE
##       FALSE, FALSE, FALSE, classif.randomForest
##  [1,]                                      TRUE
##  [2,]                                     FALSE
##  [3,]                                     FALSE
##  [4,]                                      TRUE
##  [5,]                                     FALSE
##  [6,]                                     FALSE
##  [7,]                                      TRUE
##  [8,]                                      TRUE
##  [9,]                                      TRUE
## [10,]                                      TRUE
## [11,]                                     FALSE
## [12,]                                     FALSE
## [13,]                                      TRUE
## [14,]                                     FALSE
## [15,]                                     FALSE
##       FALSE, FALSE, FALSE, classif.xgboost
##  [1,]                                 TRUE
##  [2,]                                 TRUE
##  [3,]                                 TRUE
##  [4,]                                 TRUE
##  [5,]                                FALSE
##  [6,]                                FALSE
##  [7,]                                 TRUE
##  [8,]                                 TRUE
##  [9,]                                 TRUE
## [10,]                                 TRUE
## [11,]                                FALSE
## [12,]                                FALSE
## [13,]                                 TRUE
## [14,]                                FALSE
## [15,]                                FALSE
##       FALSE, FALSE, TRUE, classif.ksvm
##  [1,]                             TRUE
##  [2,]                             TRUE
##  [3,]                             TRUE
##  [4,]                             TRUE
##  [5,]                             TRUE
##  [6,]                             TRUE
##  [7,]                            FALSE
##  [8,]                            FALSE
##  [9,]                            FALSE
## [10,]                             TRUE
## [11,]                             TRUE
## [12,]                             TRUE
## [13,]                             TRUE
## [14,]                             TRUE
## [15,]                             TRUE
##       FALSE, FALSE, TRUE, classif.randomForest
##  [1,]                                     TRUE
##  [2,]                                     TRUE
##  [3,]                                     TRUE
##  [4,]                                     TRUE
##  [5,]                                     TRUE
##  [6,]                                     TRUE
##  [7,]                                    FALSE
##  [8,]                                    FALSE
##  [9,]                                    FALSE
## [10,]                                     TRUE
## [11,]                                     TRUE
## [12,]                                     TRUE
## [13,]                                     TRUE
## [14,]                                     TRUE
## [15,]                                     TRUE
##       FALSE, FALSE, TRUE, classif.xgboost FALSE, TRUE, FALSE, classif.ksvm
##  [1,]                                TRUE                            FALSE
##  [2,]                                TRUE                            FALSE
##  [3,]                                TRUE                            FALSE
##  [4,]                                TRUE                            FALSE
##  [5,]                                TRUE                             TRUE
##  [6,]                                TRUE                             TRUE
##  [7,]                               FALSE                             TRUE
##  [8,]                               FALSE                             TRUE
##  [9,]                               FALSE                             TRUE
## [10,]                                TRUE                            FALSE
## [11,]                                TRUE                             TRUE
## [12,]                                TRUE                             TRUE
## [13,]                                TRUE                            FALSE
## [14,]                                TRUE                            FALSE
## [15,]                                TRUE                            FALSE
##       FALSE, TRUE, FALSE, classif.randomForest
##  [1,]                                     TRUE
##  [2,]                                    FALSE
##  [3,]                                    FALSE
##  [4,]                                     TRUE
##  [5,]                                    FALSE
##  [6,]                                    FALSE
##  [7,]                                     TRUE
##  [8,]                                     TRUE
##  [9,]                                     TRUE
## [10,]                                     TRUE
## [11,]                                    FALSE
## [12,]                                    FALSE
## [13,]                                     TRUE
## [14,]                                    FALSE
## [15,]                                    FALSE
##       FALSE, TRUE, FALSE, classif.xgboost
##  [1,]                                TRUE
##  [2,]                               FALSE
##  [3,]                               FALSE
##  [4,]                                TRUE
##  [5,]                               FALSE
##  [6,]                               FALSE
##  [7,]                                TRUE
##  [8,]                                TRUE
##  [9,]                                TRUE
## [10,]                                TRUE
## [11,]                               FALSE
## [12,]                               FALSE
## [13,]                                TRUE
## [14,]                               FALSE
## [15,]                               FALSE
##       SMOTE, FALSE, FALSE, classif.ksvm
##  [1,]                             FALSE
##  [2,]                             FALSE
##  [3,]                              TRUE
##  [4,]                             FALSE
##  [5,]                              TRUE
##  [6,]                              TRUE
##  [7,]                              TRUE
##  [8,]                              TRUE
##  [9,]                              TRUE
## [10,]                             FALSE
## [11,]                              TRUE
## [12,]                              TRUE
## [13,]                             FALSE
## [14,]                              TRUE
## [15,]                              TRUE
##       SMOTE, FALSE, FALSE, classif.randomForest
##  [1,]                                      TRUE
##  [2,]                                     FALSE
##  [3,]                                     FALSE
##  [4,]                                     FALSE
##  [5,]                                     FALSE
##  [6,]                                     FALSE
##  [7,]                                      TRUE
##  [8,]                                      TRUE
##  [9,]                                      TRUE
## [10,]                                     FALSE
## [11,]                                     FALSE
## [12,]                                     FALSE
## [13,]                                      TRUE
## [14,]                                     FALSE
## [15,]                                     FALSE
##       SMOTE, FALSE, FALSE, classif.xgboost
##  [1,]                                 TRUE
##  [2,]                                FALSE
##  [3,]                                FALSE
##  [4,]                                FALSE
##  [5,]                                FALSE
##  [6,]                                FALSE
##  [7,]                                 TRUE
##  [8,]                                 TRUE
##  [9,]                                 TRUE
## [10,]                                FALSE
## [11,]                                FALSE
## [12,]                                FALSE
## [13,]                                 TRUE
## [14,]                                FALSE
## [15,]                                FALSE
```

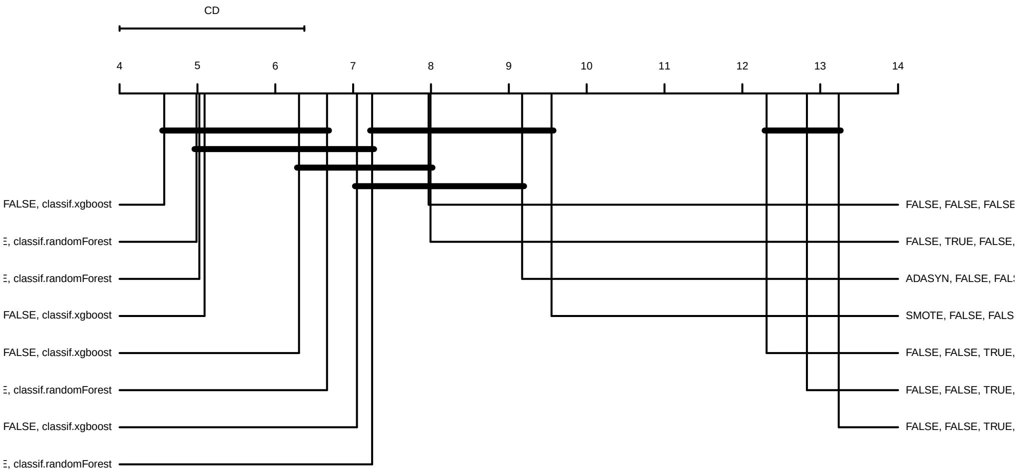
Plotando os ranks

```
print(colMeans(rankMatrix(df)))
```

```
##         ADASYN, FALSE, FALSE, classif.ksvm
##                                   9.170732
## ADASYN, FALSE, FALSE, classif.randomForest
##                                   7.243902
##      ADASYN, FALSE, FALSE, classif.xgboost
##                                   7.048780
##          FALSE, FALSE, FALSE, classif.ksvm
##                                   7.969512
##  FALSE, FALSE, FALSE, classif.randomForest
##                                   4.987805
##       FALSE, FALSE, FALSE, classif.xgboost
##                                   4.573171
##           FALSE, FALSE, TRUE, classif.ksvm
##                                  12.310976
##   FALSE, FALSE, TRUE, classif.randomForest
##                                  12.829268
##        FALSE, FALSE, TRUE, classif.xgboost
##                                  13.237805
##           FALSE, TRUE, FALSE, classif.ksvm
##                                   7.993902
##   FALSE, TRUE, FALSE, classif.randomForest
##                                   5.024390
##        FALSE, TRUE, FALSE, classif.xgboost
##                                   5.091463
##          SMOTE, FALSE, FALSE, classif.ksvm
##                                   9.548780
##  SMOTE, FALSE, FALSE, classif.randomForest
##                                   6.664634
##       SMOTE, FALSE, FALSE, classif.xgboost
##                                   6.304878
```

Plotando grafico de Critical Diference

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


CD
FALSE, classif.xgboost
E, classif.randomForest
E, classif.randomForest
FALSE, classif.xgboost
FALSE, classif.xgboost
E, classif.randomForest
FALSE, classif.xgboost
E, classif.randomForest
FALSE, FALSE, FALSE
FALSE, TRUE, FALSE,
ADASYN, FALSE, FAL
SMOTE, FALSE, FALS
FALSE, FALSE, TRUE,
FALSE, FALSE, TRUE,
FALSE, FALSE, TRUE,