keystroke

November 14, 2018

In [1]:
```
from source.util import utils as u
from source import metrics, plots
from source import handshake2, scargc, hs
import sys
import time
import os
import psutil
import resource
```

In [2]:
```
poolsize = 150
clusters = 4
n_components = 4
epsilon = 0.1
percent = 30
steps = 8
```

In [3]:
```
base = '/home/localuser/Documentos/procopio/tcc/datasets/keystroke.txt'
dataset, data_labeled, dataset_train, l_train, stream, l_stream, n_features = u.criar_da
```

Handshake

In [4]:
```
start = time.time()

predicted, updt = handshake2.handshake2(dataset, data_labeled, dataset_train, l_train, s

end = time.time()
mem = resource.getrusage(resource.RUSAGE_SELF).ru_maxrss
tempo = end - start
```

SCARGC

In [5]:
```
startScargc = time.time()

predictedS, updtS = scargc.scargc_1NN(dataset, data_labeled, dataset_train, l_train, str

endScargc = time.time()
memS = resource.getrusage(resource.RUSAGE_SELF).ru_maxrss
tempoS = endScargc - startScargc
```

```
/home/localuser/anaconda3/lib/python3.6/site-packages/sklearn/cluster/k_means_.py:896: RuntimeWa
  return_n_iter=True)
```

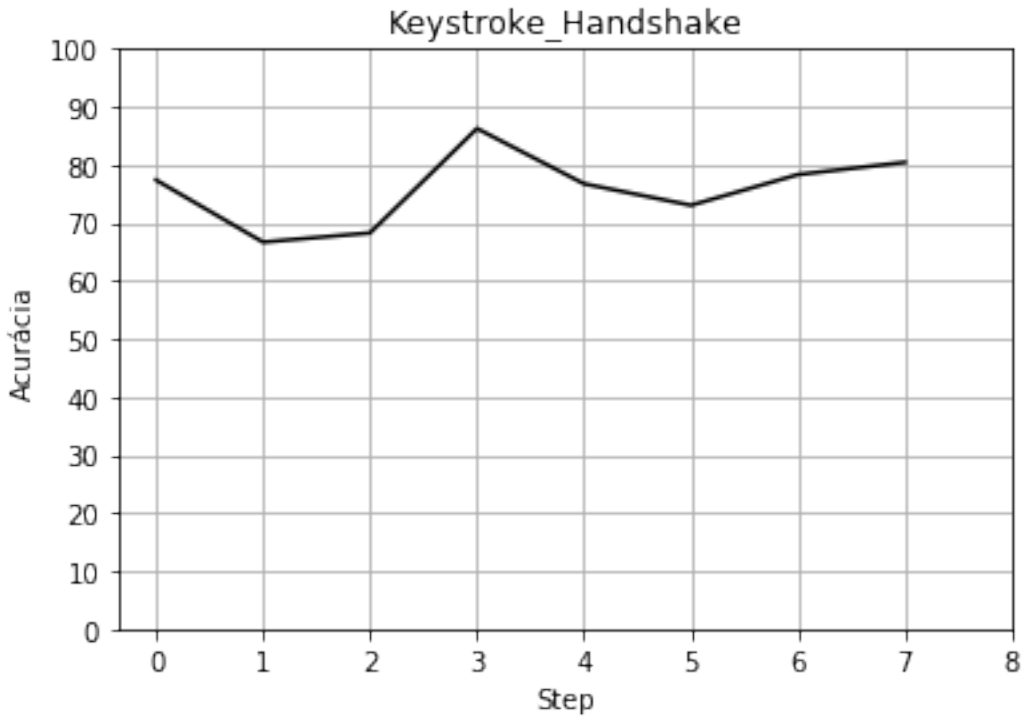
```
In [6]: acc_percent, f1_per, mcc_per = metrics.makeBatches(l_stream, predicted, len(stream), ste
        score, f1, mcc, std = metrics.metrics(acc_percent, l_stream, predicted, steps, f1_type =

        acc_percentScargc, f1_S, mcc_S = metrics.makeBatches(l_stream, predictedS, len(stream),
        scoreS, f1S, mccS, stdS = metrics.metrics(acc_percentScargc, l_stream, predictedS, steps
```

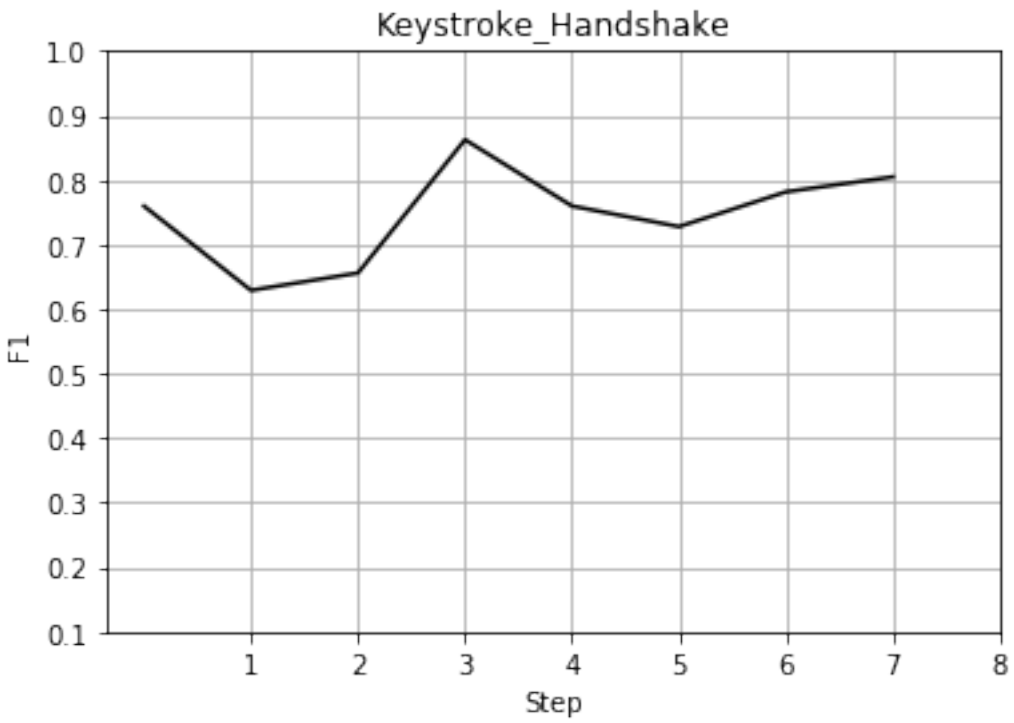
```
/home/localuser/anaconda3/lib/python3.6/site-packages/sklearn/metrics/classification.py:1135: Un
  'precision', 'predicted', average, warn_for)
```

```
In [7]: print('Tempo de Execução: ', tempo)
        print('memory peak: ', mem)
        print('Acc: ', score)
        print('Macro-F1: ', f1)
        print('MCC: ', mcc)
        print('Desvio Padrão: ', std)
        print('Numero de atualizações: ', updt)
        plots.plotAcc(acc_percent, steps, 'Keystroke_Handshake')
        plots.plotF1(f1_per, steps, 'Keystroke_Handshake')

        print('Tempo de Execução: ', tempoS)
        print('memory peak: ', memS)
        print('Acc: ', scoreS)
        print('Macro-F1: ', f1S)
        print('MCC: ', mccS)
        print('Desvio Padrão: ', stdS)
        print('Numero de atualizações: ', updtS)
        plots.plotAcc(acc_percentScargc, steps, 'Keystroke_SCARGC')
        plots.plotF1(f1_S, steps, 'Keystroke_SCARGC')

        listTime = [tempo, tempoS]
        listAcc = [score, scoreS]
        listMethod = ['Handshake', 'SCARGC']
        matrixAcc = [acc_percent[0], acc_percentScargc[0]]
        matrixF1 = [f1_per[0], f1_S[0]]

        plots.plotTime(listTime, listMethod)
        plots.plotAverageAcc(listAcc, listMethod)
        plots.plotAccuracyCurves(matrixAcc, listMethod, steps)
        plots.plotBoxplot('acc', matrixAcc, listMethod)
        plots.plotBoxplot('f1', matrixF1, listMethod)
```

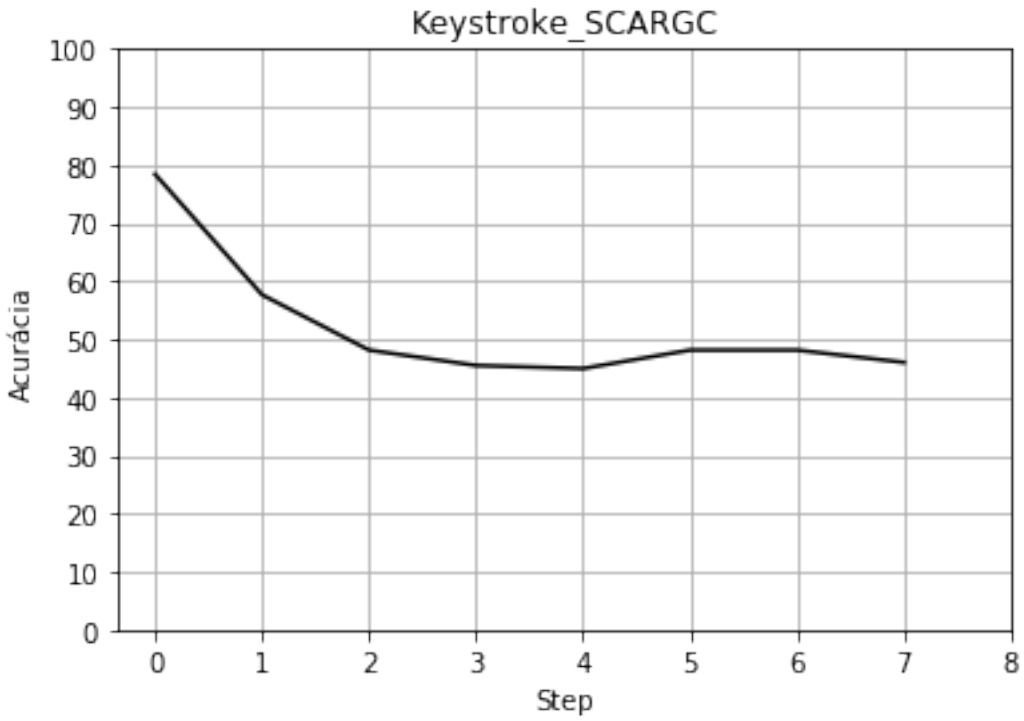
```
Tempo de Execução:  1.5982210636138916
memory peak:  129956
Acc:  0.7587475633528264
```

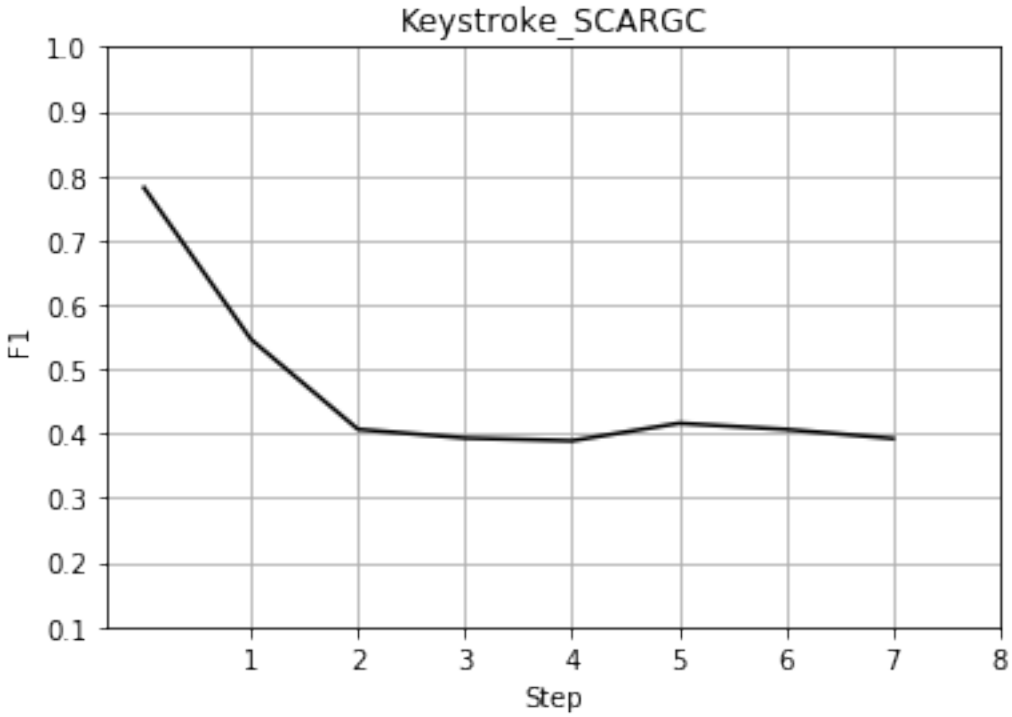
```
Macro-F1:  0.7547708246830641
MCC:  0.6866087989319529
Desvio Padrão:  0.060010006536845704
Numero de atualizações:  9
```



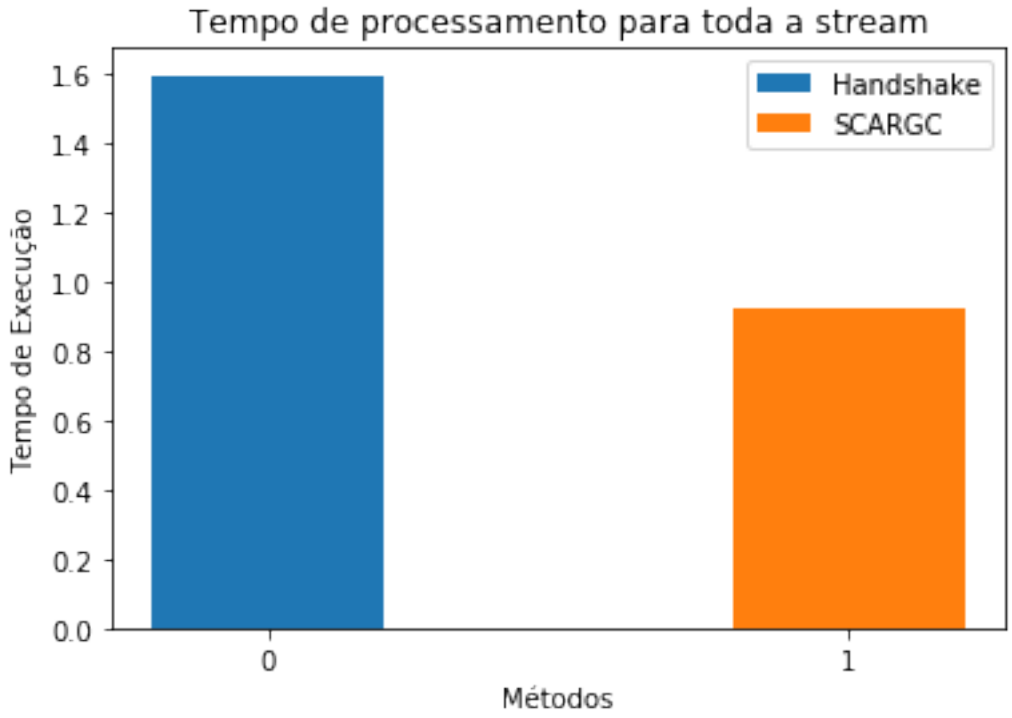


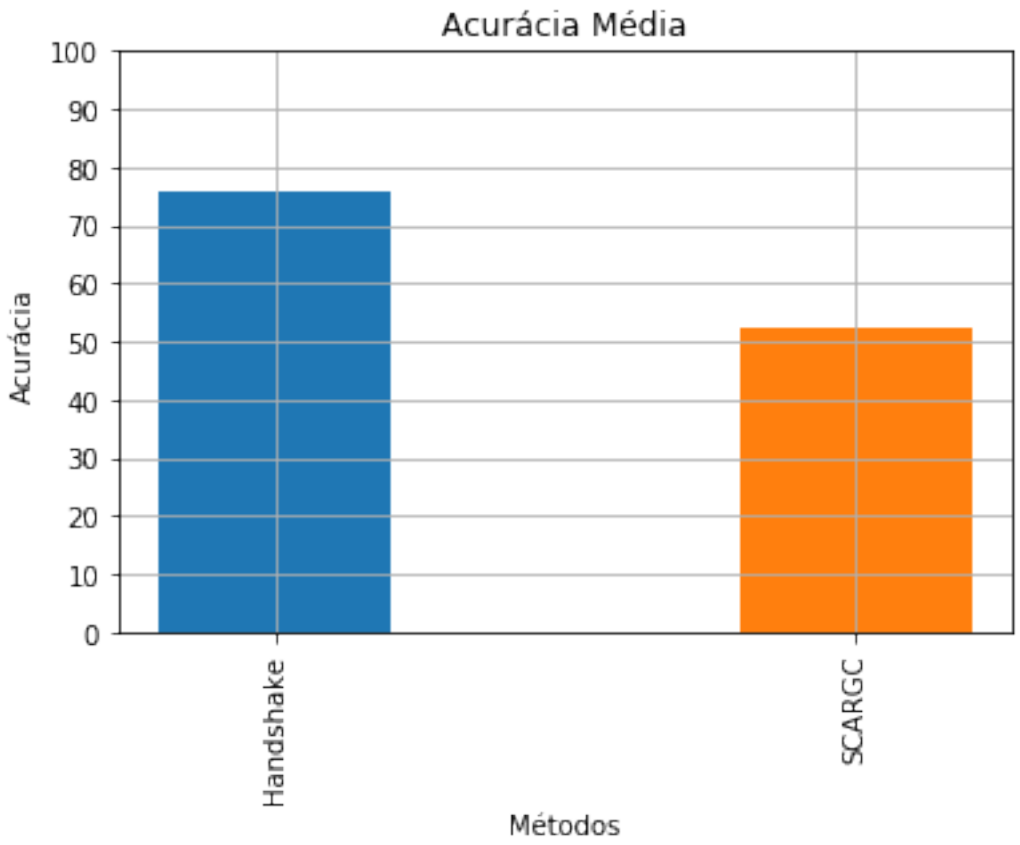

```
Tempo de Execução:  0.9226479530334473
memory peak:  130804
Acc:  0.5213067390698969
Macro-F1:  0.4838557725313432
MCC:  0.38370631569458036
Desvio Padrão:  0.10617341650761217
Numero de atualizações:  10
```

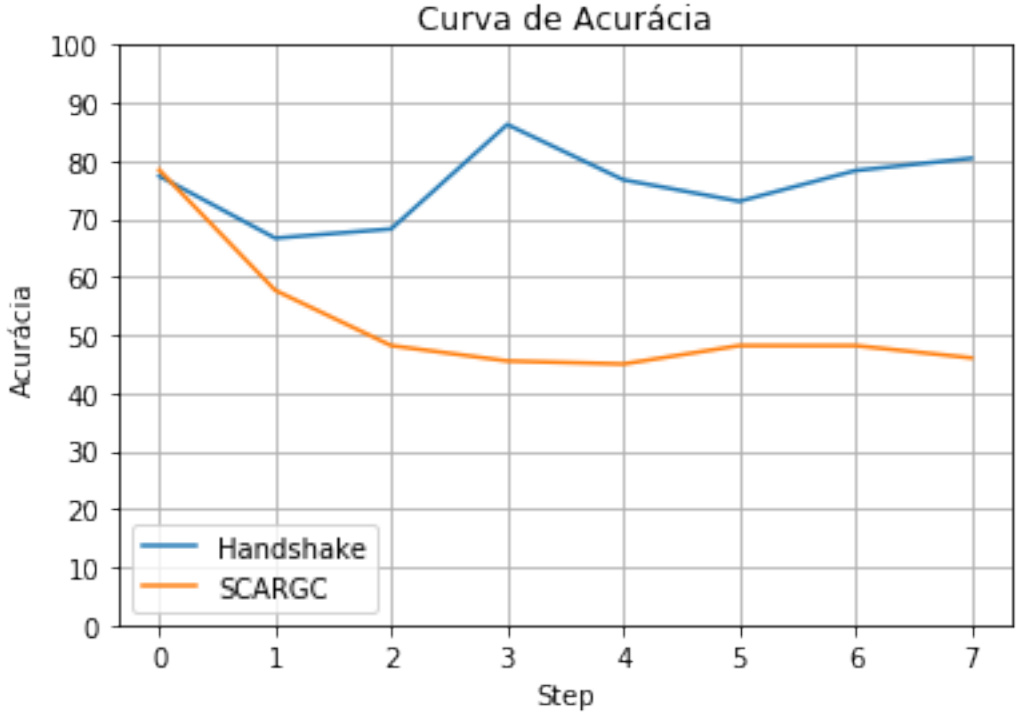

Keystroke_SCARGC
Acurácia
Step
100
90
80
70
60
50
40
30
20
10
0
0
1
2
3
4
5
6
7
8

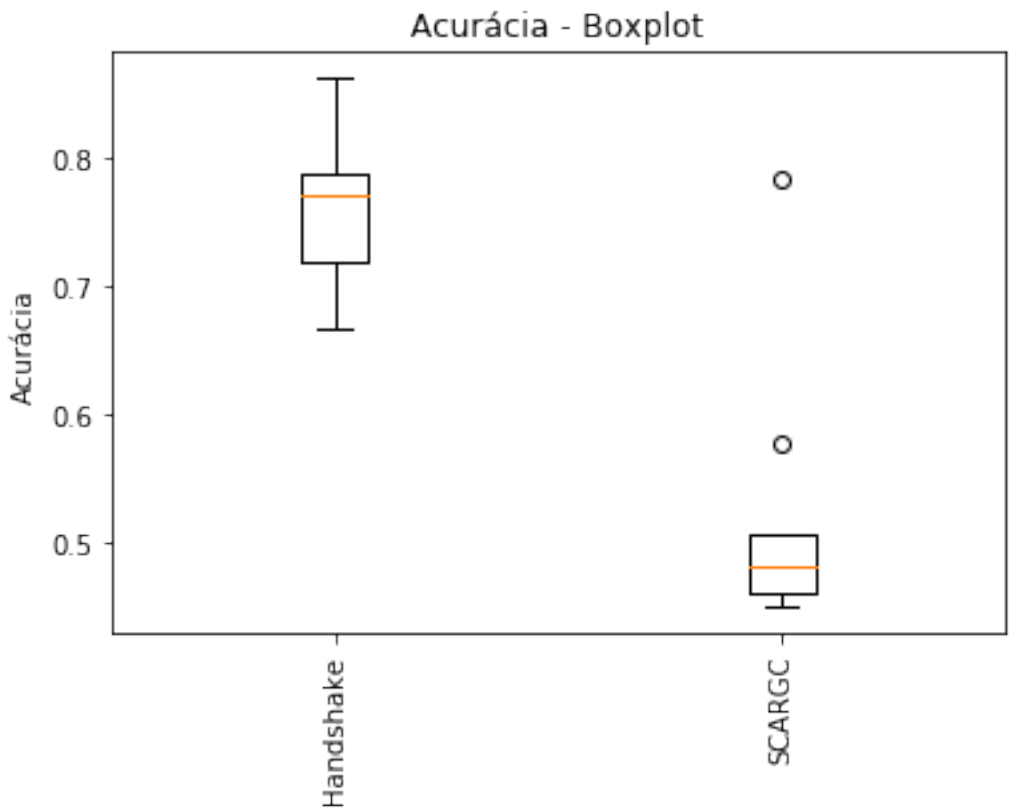

Keystroke_SCARGC
F1
Step
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
1
2
3
4
5
6
7
8

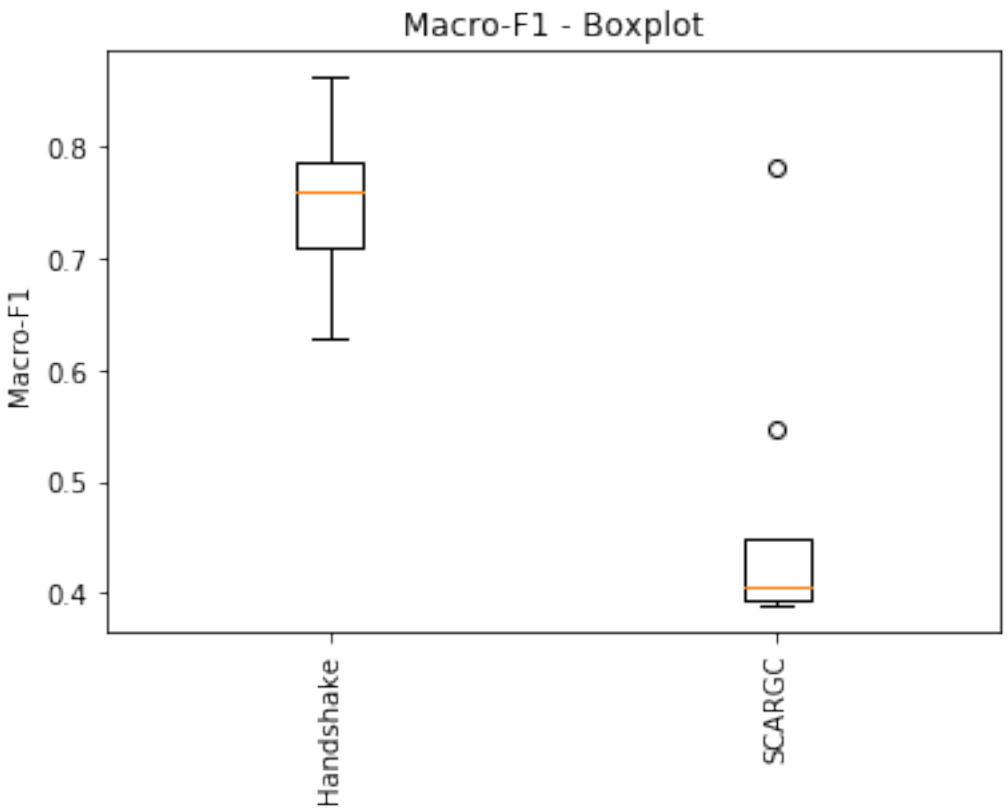
```
/home/localuser/anaconda3/lib/python3.6/site-packages/matplotlib/cbook/deprecation.py:106: Matpl
  warnings.warn(message, mplDeprecation, stacklevel=1)
```



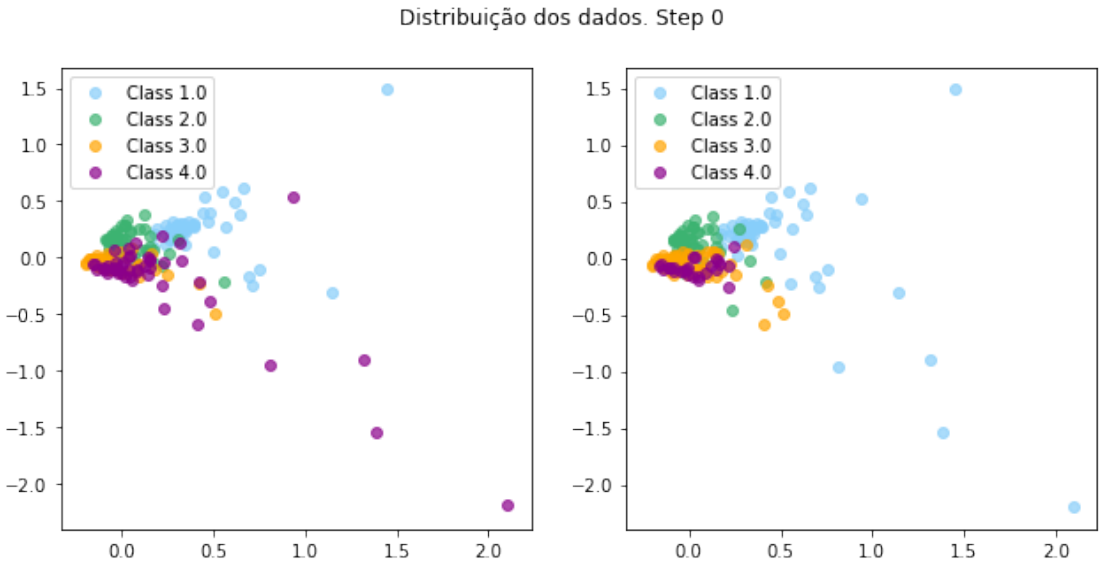

Acurácia Média
Acurácia
100
90
80
70
60
50
40
30
20
10
0
Handshake
SCARGC
Métodos


Curva de Acurácia
Acurácia
100
90
80
70
60
50
40
30
20
10
0
Handshake
SCARGC
0
1
2
3
4
5
6
7
Step

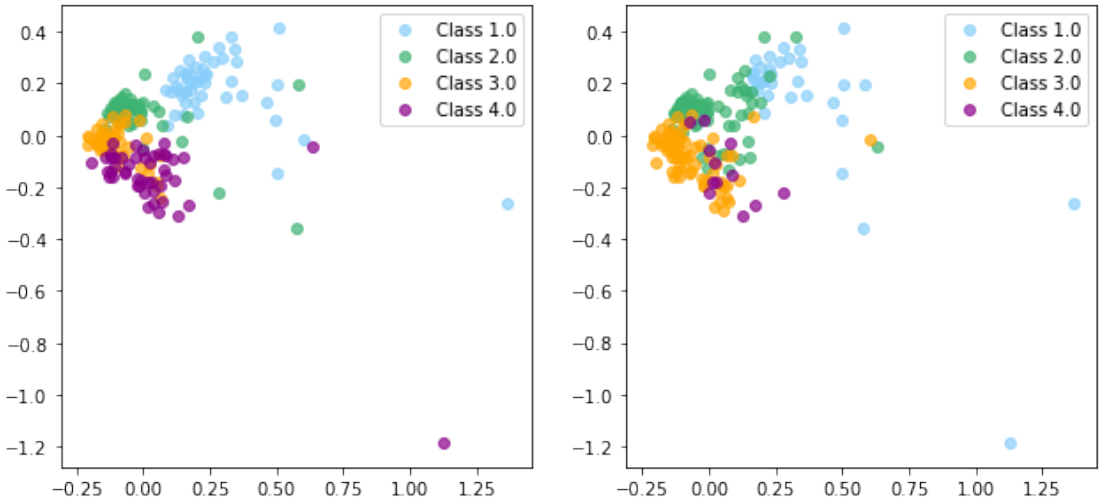

Acurácia - Boxplot
Acurácia
0.8
0.7
0.6
0.5
Handshake
SCARGC



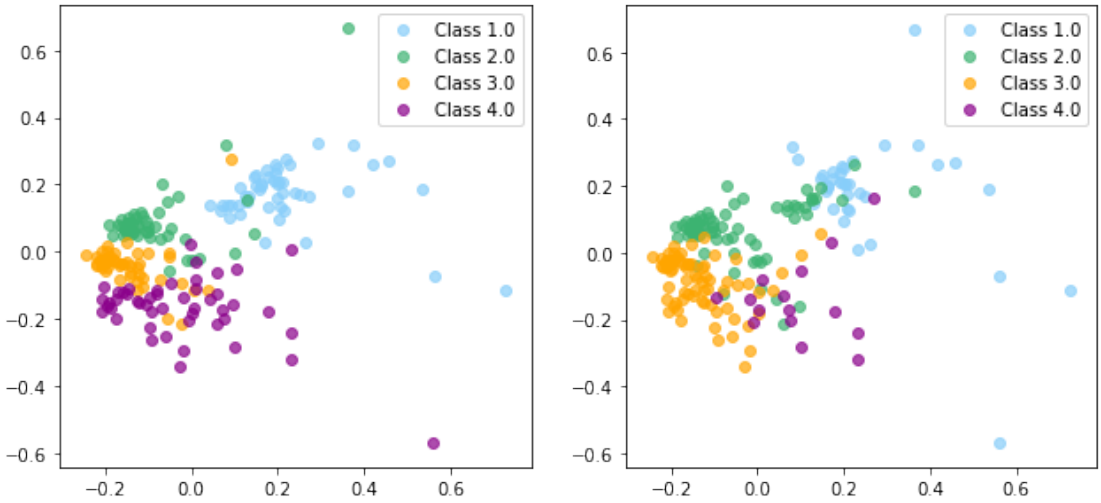

```
In [8]: plots.plotPerBatches(stream, predicted, l_stream, len(stream), steps)
```



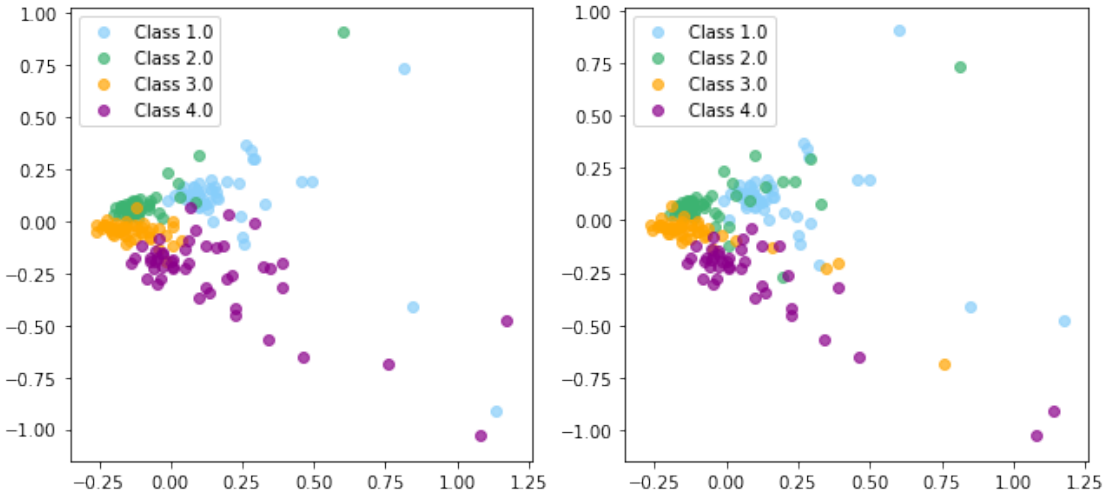
Distribuição dos dados. Step 1



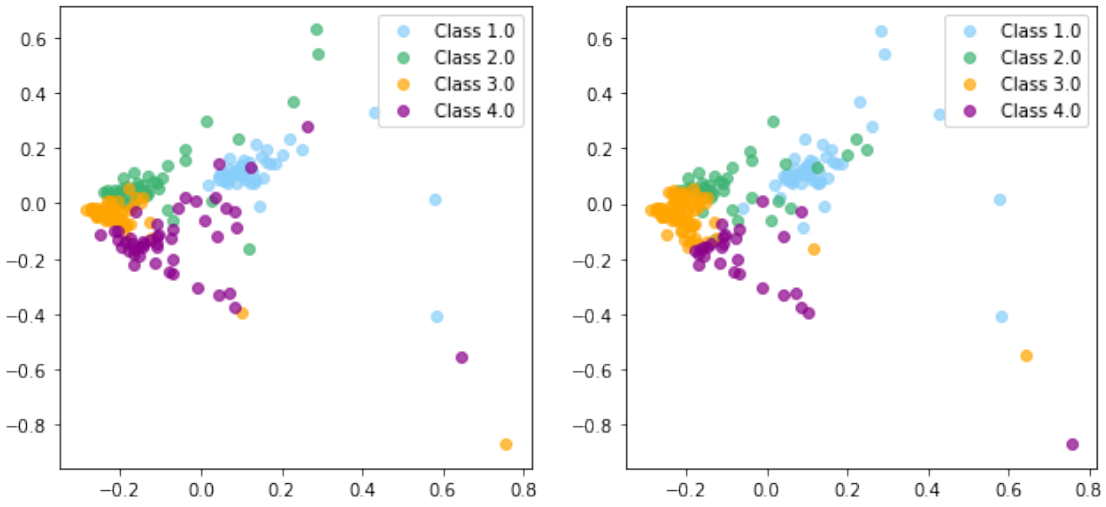
Distribuição dos dados. Step 2



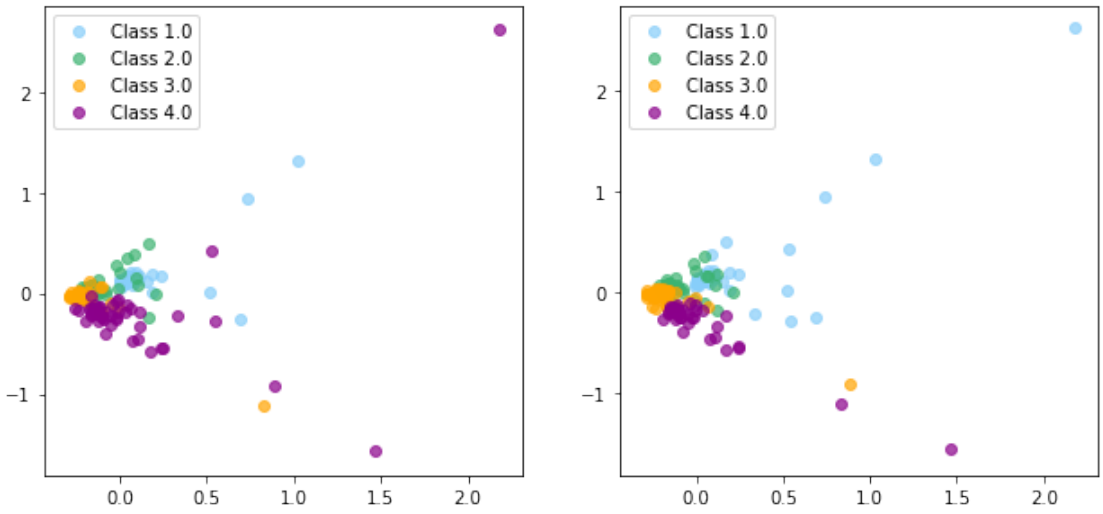
Distribuição dos dados. Step 3



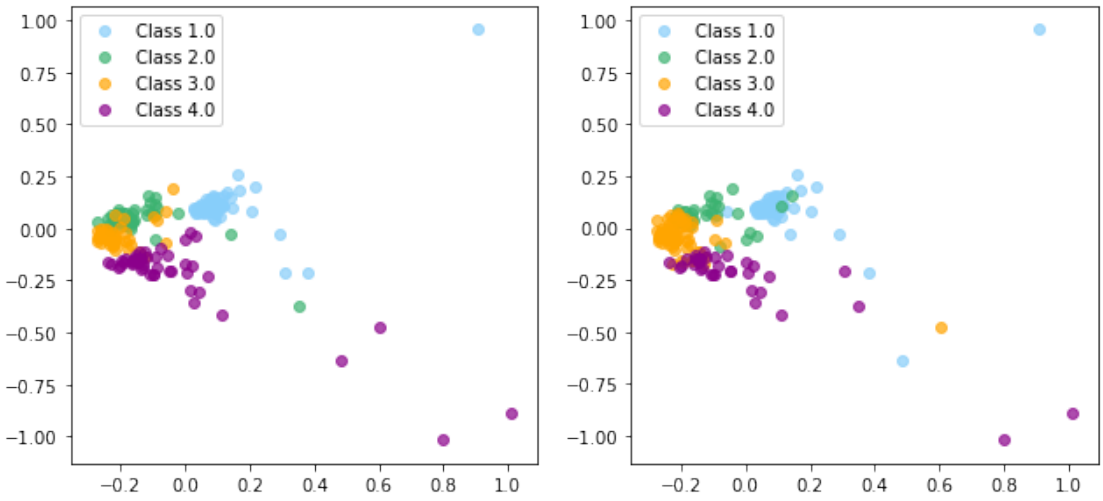
Distribuição dos dados. Step 4

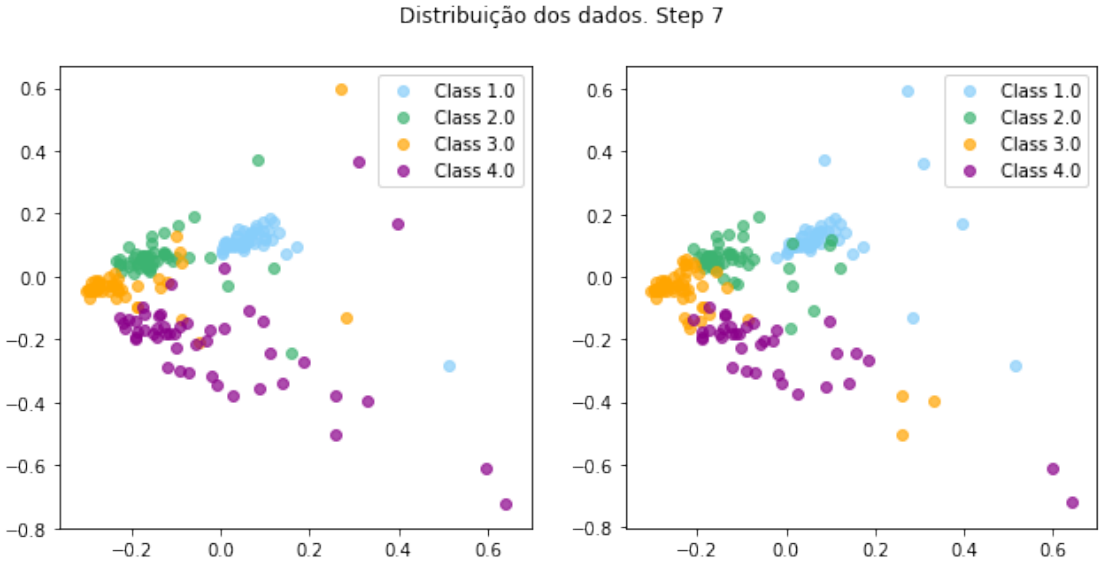


Distribuição dos dados. Step 5



Distribuição dos dados. Step 6




Distribuição dos dados. Step 7
Class 1.0
Class 2.0
Class 3.0
Class 4.0
Class 1.0
Class 2.0
Class 3.0
Class 4.0