```python
# prettyPlotConfusionMatrix.py
#
# -*- coding: utf-8 -*-
"""
plot a pretty confusion matrix with seaborn
Created on Mon Jun 25 14:17:37 2018
@author: Wagner Cipriano - wagnerbhbr - gmail - CEFETMG / MMC
Download: https://github.com/wcipriano/pretty-print-confusion-matrix
REFerences:
  https://www.mathworks.com/help/nnet/ref/plotconfusion.html
  https://stackoverflow.com/questions/28200786/how-to-plot-scikit-learn-classification-report
  https://stackoverflow.com/questions/5821125/how-to-plot-confusion-matrix-with-string-axis-rather-than-integer-in-python
  https://www.programcreek.com/python/example/96197/seaborn.heatmap
  https://stackoverflow.com/questions/19233771/sklearn-plot-confusion-matrix-with-labels/31720054
  http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html#sphx-glr-auto-examples-model-selection-plot-confusion-matrix-py
"""

#imports
from pandas import DataFrame
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.font_manager as fm
from matplotlib.collections import QuadMesh
import seaborn as sn


def get_new_fig(fn, figsize=[9,9]):
    """ Init graphics """
    fig1 = plt.figure(fn, figsize)
    ax1 = fig1.gca()   #Get Current Axis
    ax1.cla() # clear existing plot
    return fig1, ax1
#

def configcell_text_and_colors(array_df, lin, col, oText, facecolors, posi, fz, fmt, show_null_values=0):
    """
      config cell text and colors
      and return text elements to add and to dell
      @TODO: use fmt
    """
    text_add = []; text_del = [];
    cell_val = array_df[lin][col]
    tot_all = array_df[-1][-1]
    per = (float(cell_val) / tot_all) * 100
    curr_column = array_df[:,col]
    ccl = len(curr_column)

    #last line  and/or last column
    if(col == (ccl - 1)) or (lin == (ccl - 1)):
        #tots and percents
        if(cell_val != 0):
```

```
            if(col == ccl - 1) and (lin == ccl - 1):
                tot_rig = 0
                for i in range(array_df.shape[0] - 1):
                    tot_rig += array_df[i][i]
                per_ok = (float(tot_rig) / cell_val) * 100
            elif(col == ccl - 1):
                tot_rig = array_df[lin][lin]
                per_ok = (float(tot_rig) / cell_val) * 100
            elif(lin == ccl - 1):
                tot_rig = array_df[col][col]
                per_ok = (float(tot_rig) / cell_val) * 100
            per_err = 100 - per_ok
        else:
            per_ok = per_err = 0

        if per_ok ==100:
            per_ok_s ='%.2f%%'%(per_ok)
        else:
            per_ok_s = '100%'

        #text to DEL
        text_del.append(oText)

        #text to ADD
        font_prop = fm.FontProperties(weight='bold', size=fz)
        text_kwargs = dict(color='w', ha="center", va="center", gid='sum',
fontproperties=font_prop)
        lis_txt = ['%d'%(cell_val), per_ok_s, '%.2f%%'%(per_err)]
        lis_kwa = [text_kwargs]
        dic = text_kwargs.copy(); dic['color'] = 'g'; lis_kwa.append(dic);
        dic = text_kwargs.copy(); dic['color'] = 'r'; lis_kwa.append(dic);
        lis_pos = [(oText._x, oText._y-0.3), (oText._x, oText._y), (oText._x,
oText._y+0.3)]
        for i in range(len(lis_txt)):
            newText = dict(x=lis_pos[i][0], y=lis_pos[i][1], text=lis_txt[i],
kw=lis_kwa[i])
            #print 'lin: %s, col: %s, newText: %s' %(lin, col, newText)
            text_add.append(newText)
        #print '\n'

        #set background color for sum cells (last line and last column)
        carr = [0.27, 0.30, 0.27, 1.0]
        if(col == ccl - 1) and (lin == ccl - 1):
            carr = [0.17, 0.20, 0.17, 1.0]
        facecolors[posi] = carr

    else:
        if(per > 0):
            txt = '%s\n%.2f%%' %(cell_val, per)
        else:
            if(show_null_values == 0):
                txt = ''
            elif(show_null_values == 1):
                txt = '0'
            else:
                txt = '0\n0.0%'
        oText.set_text(txt)
```

```
106
107         #main diagonal
108         if(col == lin):
109             #set color of the textin the diagonal to white
110             oText.set_color('w')
111             # set background color in the diagonal to blue
112             facecolors[posi] = [0.35, 0.8, 0.55, 1.0]
113         else:
114             oText.set_color('r')
115
116     return text_add, text_del
117 #
118
119 def insert_totals(df_cm):
120     """ insert total column and line (the last ones) """
121     sum_col = []
122     for c in df_cm.columns:
123         sum_col.append( df_cm[c].sum() )
124     sum_lin = []
125     for item_line in df_cm.iterrows():
126         sum_lin.append( item_line[1].sum() )
127     df_cm['sum_lin'] = sum_lin
128     sum_col.append(np.sum(sum_lin))
129     df_cm.loc['sum_col'] = sum_col
130     #print ('\ndf_cm:\n', df_cm, '\n\b\n')
131 #
132
133 def pretty_plot_confusion_matrix(df_cm, annot=True, cmap="Oranges", fmt='.2f',
    fz=11,
134 ,       lw=0.5, cbar=False, figsize=[8,8], show_null_values=0,
    pred_val_axis='y',insertTot=True):
135     """
136       print conf matrix with default layout (like matlab)
137       params:
138         df_cm          dataframe (pandas) without totals
139         annot          print text in each cell
140         cmap           Oranges,Oranges_r,YlGnBu,Blues,RdBu, ... see:
141         fz             fontsize
142         lw             linewidth
143         pred_val_axis  where to show the prediction values (x or y axis)
144                         'col' or 'x': show predicted values in columns (x axis)
    instead lines
145                         'lin' or 'y': show predicted values in lines   (y axis)
146     """
147     if(pred_val_axis in ('col', 'x')):
148         xlbl = 'Predição'
149         ylbl = 'Classificação'
150     else:
151         xlbl = 'Classificação'
152         ylbl = 'Predição'
153         df_cm = df_cm.T
154
155     # create "Total" column
156     insert_totals(df_cm)
157
158     #this is for print allways in the same window
159     fig, ax1 = get_new_fig('Conf matrix default', figsize)
```

```
160
161     #thanks for seaborn
162     if insertTot:
163         ax = sn.heatmap(df_cm, annot=annot, annot_kws={"size": fz},
    linewidths=lw, ax=ax1,
164                         cbar=cbar, cmap=cmap, linecolor='w', fmt=fmt)
165     else:
166         ax = sn.heatmap(df_cm.iloc[:-1,:-1], annot=annot, annot_kws={"size":
    fz}, linewidths=lw, ax=ax1,
167                         cbar=cbar, cmap=cmap, linecolor='w', fmt=fmt)
168
169     #set ticklabels rotation
170     ax.set_xticklabels(ax.get_xticklabels(), rotation = 90, fontsize = 10)
171     ax.set_yticklabels(ax.get_yticklabels(), rotation = 0, fontsize = 10)
172
173     # Turn off all the ticks
174     for t in ax.xaxis.get_major_ticks():
175         t.tick1On = False
176         t.tick2On = False
177     for t in ax.yaxis.get_major_ticks():
178         t.tick1On = False
179         t.tick2On = False
180
181     #face colors list
182     quadmesh = ax.findobj(QuadMesh)[0]
183     facecolors = quadmesh.get_facecolors()
184
185     #iter in text elements
186     array_df = np.array( df_cm.to_records(index=False).tolist() )
187     text_add = []; text_del = [];
188     posi = -1 #from left to right, bottom to top.
189     for t in ax.collections[0].axes.texts: #ax.texts:
190         pos = np.array( t.get_position()) - [0.5,0.5]
191         lin = int(pos[1]); col = int(pos[0]);
192         posi += 1
193         #print ('>>> pos: %s, posi: %s, val: %s, txt: %s' %(pos, posi,
    array_df[lin][col], t.get_text()))
194
195         #set text
196         txt_res = configcell_text_and_colors(array_df, lin, col, t, facecolors,
    posi, fz, fmt, show_null_values)
197
198         text_add.extend(txt_res[0])
199         text_del.extend(txt_res[1])
200
201     #remove the old ones
202     for item in text_del:
203         item.remove()
204     #append the new ones
205     for item in text_add:
206         ax.text(item['x'], item['y'], item['text'], **item['kw'])
207
208     #titles and legends
209     ax.set_title('Matriz de Confusão',fontsize = 30)
210     ax.set_xlabel(xlbl,fontsize = 25)
211     ax.set_ylabel(ylbl,fontsize = 25)
212     plt.tight_layout()  #set layout slim
213     plt.show()
```