

```

1 # prettyPlotConfusionMatrix.py
2 #
3 # -*- coding: utf-8 -*-
4 """
5 plot a pretty confusion matrix with seaborn
6 Created on Mon Jun 25 14:17:37 2018
7 @author: Wagner Cipriano - wagnerbhbr - gmail - CEFETMG / MMC
8 Download: https://github.com/wcipriano/pretty-print-confusion-matrix
9 REferences:
10  https://www.mathworks.com/help/nnet/ref/plotconfusion.html
11  https://stackoverflow.com/questions/28200786/how-to-plot-scikit-learn-
classification-report
12  https://stackoverflow.com/questions/5821125/how-to-plot-confusion-matrix-
with-string-axis-rather-than-integer-in-python
13  https://www.programcreek.com/python/example/96197/seaborn.heatmap
14  https://stackoverflow.com/questions/19233771/sklearn-plot-confusion-matrix-
with-labels/31720054
15  http://scikit-
learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html#sphx-
glr-auto-examples-model-selection-plot-confusion-matrix-py
16 """
17
18 #imports
19 from pandas import DataFrame
20 import numpy as np
21 import matplotlib.pyplot as plt
22 import matplotlib.font_manager as fm
23 from matplotlib.collections import QuadMesh
24 import seaborn as sn
25
26
27 def get_new_fig(fn, figsize=[9,9]):
28     """ Init graphics """
29     fig1 = plt.figure(fn, figsize)
30     ax1 = fig1.gca() #Get Current Axis
31     ax1.cla() # clear existing plot
32     return fig1, ax1
33 #
34
35 def configcell_text_and_colors(array_df, lin, col, oText, facecolors, posi, fz,
fmt, show_null_values=0):
36     """
37         config cell text and colors
38         and return text elements to add and to dell
39         @TODO: use fmt
40     """
41     text_add = []; text_del = [];
42     cell_val = array_df[lin][col]
43     tot_all = array_df[-1][-1]
44     per = (float(cell_val) / tot_all) * 100
45     curr_column = array_df[:,col]
46     ccl = len(curr_column)
47
48     #last line and/or last column
49     if(col == (ccl - 1)) or (lin == (ccl - 1)):
50         #tots and percents
51         if(cell_val != 0):

```

```

52         if(col == ccl - 1) and (lin == ccl - 1):
53             tot_rig = 0
54             for i in range(array_df.shape[0] - 1):
55                 tot_rig += array_df[i][i]
56             per_ok = (float(tot_rig) / cell_val) * 100
57         elif(col == ccl - 1):
58             tot_rig = array_df[lin][lin]
59             per_ok = (float(tot_rig) / cell_val) * 100
60         elif(lin == ccl - 1):
61             tot_rig = array_df[col][col]
62             per_ok = (float(tot_rig) / cell_val) * 100
63         per_err = 100 - per_ok
64     else:
65         per_ok = per_err = 0
66
67     if per_ok == 100:
68         per_ok_s = '%.2f%%'%(per_ok)
69     else:
70         per_ok_s = '100%'
71
72     #text to DEL
73     text_del.append(oText)
74
75     #text to ADD
76     font_prop = fm.FontProperties(weight='bold', size=fz)
77     text_kwargs = dict(color='w', ha="center", va="center", gid='sum',
fontproperties=font_prop)
78     lis_txt = ['%d'%(cell_val), per_ok_s, '%.2f%%'%(per_err)]
79     lis_kwa = [text_kwargs]
80     dic = text_kwargs.copy(); dic['color'] = 'g'; lis_kwa.append(dic);
81     dic = text_kwargs.copy(); dic['color'] = 'r'; lis_kwa.append(dic);
82     lis_pos = [(oText._x, oText._y-0.3), (oText._x, oText._y), (oText._x,
oText._y+0.3)]
83     for i in range(len(lis_txt)):
84         newText = dict(x=lis_pos[i][0], y=lis_pos[i][1], text=lis_txt[i],
kw=lis_kwa[i])
85         #print 'lin: %s, col: %s, newText: %s' %(lin, col, newText)
86         text_add.append(newText)
87     #print '\n'
88
89     #set background color for sum cells (last line and last column)
90     carr = [0.27, 0.30, 0.27, 1.0]
91     if(col == ccl - 1) and (lin == ccl - 1):
92         carr = [0.17, 0.20, 0.17, 1.0]
93     facecolors[posi] = carr
94
95     else:
96         if(per > 0):
97             txt = '%s\n%.2f%%' %(cell_val, per)
98         else:
99             if(show_null_values == 0):
100                 txt = ''
101             elif(show_null_values == 1):
102                 txt = '0'
103             else:
104                 txt = '0\n0.0%'
105     oText.set_text(txt)

```

```

106
107     #main diagonal
108     if(col == lin):
109         #set color of the text in 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',
134     fz=11,
135     , lw=0.5, cbar=False, figsize=[8,8], show_null_values=0,
136     pred_val_axis='y', insertTot=True):
137     """
138     print conf matrix with default layout (like matlab)
139     params:
140     df_cm          dataframe (pandas) without totals
141     annot          print text in each cell
142     cmap           Oranges, Oranges_r, YlGnBu, Blues, RdBu, ... see:
143     fz             fontsize
144     lw            linewidth
145     pred_val_axis  where to show the prediction values (x or y axis)
146                   'col' or 'x': show predicted values in columns (x axis)
147                   'lin' or 'y': show predicted values in lines   (y axis)
148     instead lines
149
150     """
151     if(pred_val_axis in ('col', 'x')):
152         xlabel = 'Predição'
153         ylabel = 'Classificação'
154     else:
155         xlabel = 'Classificação'
156         ylabel = 'Predição'
157         df_cm = df_cm.T
158
159     # create "Total" column
160     insert_totals(df_cm)
161
162     #this is for print allways in the same window
163     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},
164 linewidths=lw, ax=ax1,
165                             cbar=cbar, cmap=cmap, linecolor='w', fmt=fmt)
166     else:
167         ax = sn.heatmap(df_cm.iloc[:-1,:-1], annot=annot, annot_kws={"size":
168 fz}, linewidths=lw, ax=ax1,
169                             cbar=cbar, cmap=cmap, linecolor='w', fmt=fmt)
170
171     #set ticklabels rotation
172     ax.set_xticklabels(ax.get_xticklabels(), rotation = 90, fontsize = 10)
173     ax.set_yticklabels(ax.get_yticklabels(), rotation = 0, fontsize = 10)
174
175     # Turn off all the ticks
176     for t in ax.xaxis.get_major_ticks():
177         t.tick10n = False
178         t.tick20n = False
179     for t in ax.yaxis.get_major_ticks():
180         t.tick10n = False
181         t.tick20n = False
182
183     #face colors list
184     quadmesh = ax.findobj(QuadMesh)[0]
185     facecolors = quadmesh.get_facecolors()
186
187     #iter in text elements
188     array_df = np.array( df_cm.to_records(index=False).tolist() )
189     text_add = []; text_del = [];
190     posi = -1 #from left to right, bottom to top.
191     for t in ax.collections[0].axes.texts: #ax.texts:
192         pos = np.array( t.get_position() ) - [0.5,0.5]
193         lin = int(pos[1]); col = int(pos[0]);
194         posi += 1
195         #print ('>>> pos: %s, posi: %s, val: %s, txt: %s' %(pos, posi,
196 array_df[lin][col], t.get_text()))
197
198     #set text
199     txt_res = configcell_text_and_colors(array_df, lin, col, t, facecolors,
200 posi, fz, fmt, show_null_values)
201
202     text_add.extend(txt_res[0])
203     text_del.extend(txt_res[1])
204
205     #remove the old ones
206     for item in text_del:
207         item.remove()
208
209     #append the new ones
210     for item in text_add:
211         ax.text(item['x'], item['y'], item['text'], **item['kw'])
212
213     #titles and legends
214     ax.set_title('Matriz de Confusão',fontsize = 30)
215     ax.set_xlabel(xlbl,fontsize = 25)
216     ax.set_ylabel(ylbl,fontsize = 25)
217     plt.tight_layout() #set layout slim
218     plt.show()

```

