```
# -*- coding: utf-8 -*-
"""TCC_CaioMota_09/06/2021_AnaliseExploratória_BaseBrasil_2016ate2018.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/10IT-RG6DPcliZ2bLMvFicESEuH2DRB_P

**Implementação do algoritmo de Analise Exploratoria **
<br>
<br>
**Autor**: Caio Augusto de Souza Mota (*caiomota802@gmail.com*)

Data: 09/06/2021

**Revisor**: Carlos Eduardo Beluzo (*cbeluzo@gmail.com*)
<br>
<br>
*Codigo adaptado de Baligh Mnassri disponivel em: https://www.kaggle.com/mnassrib/titanic-logistic-regression-with-python/notebook*

---
Este código é parte do Trabalho de Conclusão de Curso  apresentado como exigência parcial para obtenção do diploma do Curso de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo Câmpus Campinas.

# 1. Importação de Bibliotecas, carga de dados e funções
"""

# instalando o Synapse Client
! pip install synapseclient
```

```
29
30 import os
31 import synapseclient as syna
32 from getpass import getpass
33
34 import numpy as np
35 from math import sqrt
36 import pandas as pd
37
38
39 import matplotlib.pyplot as plt
40 plt.rc("font", size=14)
41
42 from  matplotlib.ticker import PercentFormatter
43 import matplotlib.ticker as ticker
44
45 import seaborn as sns
46 sns.set(style="white") #white background style for
   seaborn plots
47 sns.set(style="whitegrid", color_codes=True)
48
49 """## 1.1 Carregando base de dados disponícel no
   Synapse"""
50
51 # BRNeodeath
52 # Recuperando a base de dados do repositório de dados
    Synapse
53 syn = syna.Synapse()
54 syn.login(input('Sybapse User: '), getpass('Passwd:'
   ))
55
56 # Obtendo um ponteiro e baixando os dados
57 dataset = syn.get(entity='syn25575811') # ID do
   dataset BRNeodeath
58
59 df_ori = pd.read_csv(dataset.path)
60 df_ori
61
62 df = df_ori.sample(frac=1).copy()
63 df.shape
64
```

```
"""# 2. Etapa de pre-processamento de dados

Esta base já foi pre-processada, logo não precisa de
 tratamento de nulos. Apenas referencie ela dizendo
que você recuperou ela do Synapse por link msm. Diz
que ela não é pública, mas pode ser reconstruída a
partir de dados públicos do SIM e SINASC.

O único campo com valores nulos é "death_date", que
os registros dos vivos, logo o campo deve ser nulo.
De qualquer forma liste as features que serão
removidas.

birth_date                          0
death_date                    6719191
birth_year                          0
uf                                  0
id                                  0

Não precisa colocar código no TCC, apenas descritivo
.

## Não incluir
"""

# verificar os valores ausentes nos dados de treino
df.isnull().sum()

# Base final
features = ['maternal_age', 'tp_maternal_schooling'
, 'tp_marital_status',
                'tp_maternal_race', 'num_live_births'
,'num_fetal_losses',
                'num_previous_gestations', '
num_normal_labors', 'num_cesarean_labor',
                'tp_pregnancy', 'newborn_weight', '
gestaional_week',
                'cd_apgar1', 'cd_apgar5', '
has_congenital_malformation',
                'tp_newborn_presentation', '
num_prenatal_appointments', 'tp_labor',
```

```
 92                 'was_cesarean_before_labor', '
    was_labor_induced', 'tp_childbirth_care',
 93                 'tp_robson_group','is_neonatal_death'
    ]
 94
 95 df =  df[features]
 96
 97 # Exibindo o nome de todas as colunas
 98 colunas = df.columns
 99 colunas
100
101 # Exibindo os valores de cada coluna (domínio de
    valores)
102 for col in colunas:
103     print(col, ": ", df[col].dtype)
104     print(df[col].unique(), "\n")
105
106 df.is_neonatal_death.value_counts()
107
108 """## Incluir estas 4 tabelas como apendice, e citar
     no início da seção que estão disponíveis as tabelas
     sumário da base nos apendices XX, X1, X3 e X4.
109
110 ### Características demográficas e socioeconômicas
    maternas
111 """
112
113 aux = df[['maternal_age', 'tp_maternal_schooling', '
    tp_marital_status', 'tp_maternal_race']].describe()
114 aux.to_csv("tab1.csv")
115 aux
116
117 """### Variáveis obstétricas maternas"""
118
119 aux = df[['num_live_births','num_fetal_losses', '
    num_previous_gestations',
120     'num_normal_labors', 'num_cesarean_labor','
    tp_pregnancy']].describe()
121 aux.to_csv("tab2.csv")
122 aux
123
```

```
"""### Variáveis de histórico de gravidez"""

aux = df[['num_prenatal_appointments', 'tp_labor', 'tp_robson_group']].describe()
aux.to_csv("tab3.csv")
aux

"""### Variáveis relacionadas ao recém-nascido"""

aux = df[['newborn_weight', 'gestaional_week',
                'cd_apgar1', 'cd_apgar5', 'has_congenital_malformation',
                'tp_newborn_presentation', 'tp_labor',
                'was_cesarean_before_labor', 'was_labor_induced', 'tp_childbirth_care'
                ,'is_neonatal_death']].describe()
aux.to_csv("tab4.csv")
aux

"""# 3. Etápa de Análise Exploratória

No grafico de pizza abaixo podemos ver os valores que temos em nosso rotulo sendo 99.4% dos dados de nascidos vivos e 0.6% de nascidos mortos, isso mostra para nos o quanto a base de dados esta desbalanceada.
"""

fig,ax = plt.subplots(figsize=(6,6),subplot_kw=dict(aspect="equal"))

topic = ['Nascidos Vivos','Nascidos Mortos']
labels = list(topic)

data = [df['is_neonatal_death'].value_counts()]

def func(pct,allvals):
  absolute = int(pct/100.*np.sum(allvals))
  return "{:.1f}%\n({:d})".format(pct,absolute)

```

```
156 wedges,texts,autotexts = ax.pie(data,autopct=lambda
    pct: func(pct, data),textprops=dict(color="w"))
157
158 ax.legend(wedges,labels,title="DataFrames",loc= "
    center left",bbox_to_anchor=(1,0,0.5,1))
159
160 plt.setp(autotexts,size=12,weight="bold")
161
162 ax.set_title("Porcentagem de amostras de Nascidos
    Vivos e Nascidos Mortos")
163 plt.show
164
165 """<br>Nessa sessão plotamos graficos que nos ajuda
    a visualizar a distribuição que temos nos dados,
    vendo quais são as medias de cada variáveis.
166
167 No boxploit abaixo a variável utilizada foi "Peso ao
     nascer". Como podemos observar no peso ao nascer
    majoritariamente esta entre 3000 e 3500 gramas e com
     variáção média de 2000 a 4500 gramas. A média do
    peso ao nascer dos recem nascidos é de 3187 gramas.
168 """
169
170 # Função pra imprimir boxplot
171 def print_boxplot(x_, y_, h_, lbl):
172   legendas = ["Vivos", #0
173               "Mortos", #1
174               ]
175   ax = sns.boxplot(x=x_, y=y_,  hue=h_, data=df)
176   ax.set(xlabel='')
177   ax.set(ylabel=lbl)
178   ax.set_title('Distribuição da variável "%s"' % lbl
    )
179   ax.set_xticklabels(['Vivos', 'Mortos'])
180
181   h, l = ax.get_legend_handles_labels()
182   ax.legend(h, legendas)
183
184   ax.figure.savefig("y_.png")
185
186 print_boxplot('is_neonatal_death', 'newborn_weight'
```

```
186 , 'is_neonatal_death', 'peso ao nascer')
187
188 print_boxplot('is_neonatal_death', 'maternal_age', '
    is_neonatal_death', 'idade da mãe')
189
190 print_boxplot('is_neonatal_death', 'gestaional_week'
    , 'is_neonatal_death', 'semanas gestacionais')
191
192 legendas = ["Vivos", #0
193             "Mortos", #1
194             ]
195
196 ax = sns.histplot(data=df, x="newborn_weight", hue="
    is_neonatal_death", kde=True,
197                       )
198 ax.set_yscale('log')
199
200 ax.set(xlabel='Peso ao nascer')
201 ax.set(ylabel='Distribuição (log)')
202 ax.set_title('Distribuição da variável "%s" por
    classes' % "peso ao nascer")
203
204 ax.legend(legendas)
205
206 ax.figure.savefig("ditro.png")
207
208 """Nesse Grafico temos a distribuição dos dados da
    feature "anos de escolaridade da mãe", essa feature
    utiliza dados categoricos Nominais as categorias sao
    :
209 <br>1 - nenhum;
210 <br>2 - de 1 a 3 anos;
211 <br>3 - de 4 a 7 anos;
212 <br>4 - de 8 a 11 anos;
213 <br>5 - 12 e mais anos;
214 <br>9 - ignorado;
215 """
216
217 plt.figure(figsize=(10,4))
218 titulo = "Distribuição da variável Escolaridade da
    mãe"
```

```
219 label_Y = "Escolaridade da mãe"
220 label_X = "Contagem"
221 legendas = ["Dados errados",
222              "nenhum", #1
223              "de 1 a 3 anos", #2
224              "de 4 a 7 anos",#3
225              "de 8 a 11 anos", #4
226              "12 e mais anos",#6
227              "ignorado"] #9
228
229 ax = sns.countplot(y = 'tp_maternal_schooling', hue=
    'tp_maternal_schooling', dodge=False, data = df)
230 ax.set(xlabel=label_X)
231 ax.set(ylabel=label_Y)
232 ax.set_title(titulo)
233 ax.legend(loc='upper right', labels=legendas)
234 ax.figure.savefig("tp_maternal_schooling.png")
235
236 # media Semana de gestação(por intervalos)
237 print('A média de "Anos de escolaridade da mãe" é %.
    0f' %(df["tp_maternal_schooling"].mean(skipna=True
    )))
238 # mediana Semana de gestação(por intervalos)
239 print('A mediana de  "Anos de escolaridade da mãe" é
     %.0f' %(df["tp_maternal_schooling"].median(skipna=
    True)))
240
241 print('Agrupamento dos anos de escolaridade da mãe:'
    )
242 print(df['tp_maternal_schooling'].value_counts())
243
244 variável = 'num_live_births'
245 plt.figure(figsize=(10,4))
246 titulo = "Distribuição da variável Número de
    nascidos vivos"
247 label_X = "Número de nascidos vivos"
248 label_Y = "Contagem"
249
250 ax = sns.countplot(x = 'num_live_births', dodge=
    False, data = df)
251 ax.set(xlabel=label_X)
```

```
252 ax.set(ylabel=label_Y)
253 ax.set_title(titulo)
254 ax.figure.savefig("num_live_births.png")
255
256 # media Semana de gestação(por intervalos)
257 print('A média de "Número de nascidos vivos da mãe"
    é %.1f' %(df["num_live_births"].mean(skipna=True)))
258 # mediana Semana de gestação(por intervalos)
259 print('A mediana de  "Número de nascidos vivos da
    mãe" é %.0f' %(df["num_live_births"].median(skipna=
    True)))
260
261 print('Agrupamento do número de nascidos vivos da
    mãe:')
262 print(df['num_live_births'].value_counts())
263
264 variavel = 'cd_apgar1'
265 plt.figure(figsize=(10,4))
266 titulo = "Distribuição da variável Pontuação de
    Apgar de 1 minuto"
267 label_X =  "Pontuação de Apgar de 1 minuto"
268 label_Y = "Contagem"
269
270
271 ax = sns.countplot(x=variavel, dodge=False, data =
    df)
272 ax.set(xlabel=label_X)
273 ax.set(ylabel=label_Y)
274 ax.set_title(titulo)
275 ax.figure.savefig("graf.png")
276
277 # media Semana de gestação(por intervalos)
278 print('A média de "Pontuação de Apgar de 1 minuto" é
     %.0f' %(df["cd_apgar1"].mean(skipna=True)))
279 # mediana Semana de gestação(por intervalos)
280 print('A mediana de  "Pontuação de Apgar de 1 minuto
    " é %.0f' %(df["cd_apgar1"].median(skipna=True)))
281
282 print('Agrupamento da pontuação de Apgar de 1 minuto
    :')
283 print(df['cd_apgar1'].value_counts())
```

```
284
285 variavel = 'cd_apgar5'
286 plt.figure(figsize=(10,4))
287 titulo = "Distribuição da variável Pontuação de
    Apgar de 5 minuto"
288 label_X = "Pontuação de Apgar de 5 minuto"
289 label_Y = "Contagem"
290
291 ax = sns.countplot(x=variavel, dodge=False, data =
    df)
292 ax.set(xlabel=label_X)
293 ax.set(ylabel=label_Y)
294 ax.set_title(titulo)
295 ax.figure.savefig("graf.png")
296
297 # media Semana de gestação(por intervalos)
298 print('A média de "Pontuação de Apgar de 5 minuto" é
     %.0f' %(df["cd_apgar5"].mean(skipna=True)))
299 # mediana Semana de gestação(por intervalos)
300 print('A mediana de  "Pontuação de Apgar de 5 minuto
    " é %.0f' %(df["cd_apgar5"].median(skipna=True)))
301
302 print('Agrupamento da pontuação de Apgar de 5 minuto
    :')
303 print(df['cd_apgar5'].value_counts())
304
305 variavel = 'has_congenital_malformation'
306 plt.figure(figsize=(10,4))
307 titulo = "Distribuição da variável Presença de
    malformação congênita"
308 label_Y = "Presença de malformação congênita"
309 label_X = "Contagem"
310 legendas = ["Sim", #1
311             "Não", #2
312             "ignorado"] #9
313
314 ax = sns.countplot(y=variavel, hue=variavel, dodge=
    False, data = df)
315 ax.set(xlabel=label_X)
316 ax.set(ylabel=label_Y)
317 ax.set_title(titulo)
```

```
318 ax.legend(loc='upper right', labels=legendas)
319 ax.figure.savefig("graf.png")
320
321 # media Semana de gestação(por intervalos)
322 print('A média de "Presença de malformação congênita
    " é %.0f' %(df["has_congenital_malformation"].mean(
    skipna=True)))
323 # mediana Semana de gestação(por intervalos)
324 print('A mediana de  "Presença de malformação
    congênita" é %.0f' %(df["has_congenital_malformation
    "].median(skipna=True)))
325
326 print('Agrupamento da Presença de malformação
    congênita:')
327 print(df['has_congenital_malformation'].value_counts
    ())
328
329 variavel = 'num_previous_gestations'
330 plt.figure(figsize=(10,4))
331 titulo = "Distribuição da variável Número de
    gestações anteriores"
332 label_X = "Número de gestações anteriores"
333 label_Y = "Contagem"
334
335 ax = sns.countplot(x=variavel, dodge=False, data =
    df)
336 ax.set(xlabel=label_X)
337 ax.set(ylabel=label_Y)
338 ax.set_title(titulo)
339 ax.figure.savefig("graf.png")
340
341 # media Semana de gestação(por intervalos)
342 print('A média de "Número de gestações anteriores" é
     %.0f' %(df["num_previous_gestations"].mean(skipna=
    True)))
343 # mediana Semana de gestação(por intervalos)
344 print('A mediana de  "Número de gestações anteriores
    " é %.0f' %(df["num_previous_gestations"].median(
    skipna=True)))
345
346 print('Agrupamento do Número de gestações anteriores
```

```
346  da mãe:')
347 print(df['num_previous_gestations'].value_counts())
348
349 variavel = 'tp_childbirth_care'
350 plt.figure(figsize=(10,4))
351 titulo = "Distribuição da variável Assistência ao
    parto"
352 label_Y = "Assistência ao parto"
353 label_X = "Contagem"
354 legendas = ["nenhum", #0
355              "Doutor", #1
356              "Enfermeira ou obstetra",#2
357              "Parteira", #3
358              "Outros",#4
359              "Ignorado"] #9
360
361 ax = sns.countplot(y=variavel, hue=variavel, dodge=
    False, data = df)
362 ax.set(xlabel=label_X)
363 ax.set(ylabel=label_Y)
364 ax.set_title(titulo)
365 ax.legend(loc='upper right', labels=legendas)
366 ax.figure.savefig("graf.png")
367
368 # media Semana de gestação(por intervalos)
369 print('A média de "Assistência ao parto" é %.0f' %(
    df["tp_childbirth_care"].mean(skipna=True)))
370 # mediana Semana de gestação(por intervalos)
371 print('A mediana de  "Assistência ao parto" é %.0f'
     %(df["tp_childbirth_care"].median(skipna=True)))
372
373 print('Agrupamento do Assistência ao parto:')
374 print(df['tp_childbirth_care'].value_counts())
375
376 plt.figure(figsize=(15,8))
377 ax = sns.kdeplot(df["newborn_weight"][df.
    is_neonatal_death == 1], color="lightcoral", shade=
    True)
378 sns.kdeplot(df["newborn_weight"][df.
    is_neonatal_death == 0], color="darkturquoise",
    shade=True)
```

```
379 plt.legend(['Mortos', 'Vivos'])
380 plt.title('Gráfico de densidade de Peso ao nascer em
     gramas', fontsize=15)
381 ax.set(xlabel='Peso ao nascer em gramas')
382 ax.set(ylabel='Densidade')
383 ax.tick_params(labelsize=15)
384 plt.xlim(0,6000)
385 plt.show()
386
387 X = df[['maternal_age',
388        'num_prenatal_appointments',
389        'cd_apgar1' ,
390        'cd_apgar5',
391        'newborn_weight',
392        'num_previous_gestations',
393        'num_normal_labors',
394        'num_cesarean_labor',
395        'gestaional_week',
396        'tp_newborn_presentation']]
397
398 X.rename(columns = {'maternal_age':'Idade da Mãe',
399                     'num_prenatal_appointments':'
    Número de consultas de pré-natal por faixas'
400                     ,'cd_apgar1':'Pontuação de Apgar
     de 1 minuto' ,
401                     'cd_apgar5':'Pontuação de Apgar
    de 5 minuto',
402                     'newborn_weight':'Peso ao nascer
     em gramas',
403
404                     'num_previous_gestations':'
    Número de gestações anteriores',
405                     'num_normal_labors':'Número de
    partos normais (trabalhos de parto)',
406                     'num_cesarean_labor':'Número de
    partos cesáreos (partos)'
407                     ,'gestaional_week':'Semana de
    gestação (por intervalos)',
408                     'tp_newborn_presentation':'Tipo
    de apresentação de recém-nascido'
409                     }, inplace=True)
```

```
410
411 plt.subplots(figsize=(15, 10))
412 plt.tick_params(labelsize=12)
413 sns.heatmap(X.corr(), annot=True, cmap="RdYlGn")
414 plt.show()
415
416 """# Distribuição por raça"""
417
418 ax = sns.boxplot(y="maternal_age", x="
    tp_maternal_race", data=df)
419 ax.set(xlabel="Raça da mãe")
420 ax.set(ylabel="Idade da mãe")
421
422 ax = sns.boxplot(y="newborn_weight", x="
    tp_maternal_race", data=df)
423 ax.set(xlabel="Raça da mãe")
424 ax.set(ylabel="Peso ao nascer")
425
426 ax=sns.boxplot(y="gestaional_week", x="
    tp_maternal_race", data=df)
427 ax.set(xlabel="Raça da mãe")
428 ax.set(ylabel="Semanas de gestação")
429
430 """# Distribuição por estado civil"""
431
432 ax=sns.boxplot(y="newborn_weight", x="
    tp_marital_status", data=df)
433 ax.set(xlabel="Estado civil da mãe")
434 ax.set(ylabel="Peso ao nascer")
435
436 ax=sns.boxplot(y="gestaional_week", x="
    tp_marital_status", data=df)
437 ax.set(xlabel="Estado civil da mãe")
438 ax.set(ylabel="Semanas de gestação")
439
440 ax=sns.boxplot(y="maternal_age", x="
    tp_marital_status", data=df)
441 ax.set(xlabel="Estado civil da mãe")
442 ax.set(ylabel="Idade da mãe")
443
444 """# Distribuição por tp_maternal_schooling"""
```

```
445
446 ax=sns.boxplot(y="maternal_age", x="
    tp_maternal_schooling", data=df)
447 ax.set(xlabel="Escolaridade da mãe")
448 ax.set(ylabel="Idade da mãe")
449
450 ax=sns.boxplot(y="newborn_weight", x="
    tp_maternal_schooling", data=df)
451 ax.set(xlabel="Escolaridade da mãe")
452 ax.set(ylabel="Peso ao nascer")
453
454 ax=sns.boxplot(y="gestaional_week", x="
    tp_maternal_schooling", data=df)
455 ax.set(xlabel="Escolaridade da mãe")
456 ax.set(ylabel="Semanas de gestação")
```