Neste notebook, trataremos o dataset que será submetido ao ML

In [1]:

```
import pandas as pd
from collections import Counter
```

In [2]:

```
df = pd.read_csv(r'C:\Users\73594253368\Desktop\Curso\Datasets\Procon\df_merged.csv')
```

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16028 entries, 0 to 16027
Data columns (total 27 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Unnamed: 0             16028 non-null  int64  
 1   index1                 16028 non-null  int64  
 2   AnoCalendario          16028 non-null  int64  
 3   DataArquivamento       16028 non-null  object 
 4   DataAbertura           16028 non-null  object 
 5   CodigoRegiao           16028 non-null  int64  
 6   Regiao                 16028 non-null  object 
 7   UF                     16028 non-null  object 
 8   strRazaoSocial         16028 non-null  object 
 9   strNomeFantasia        13123 non-null  object 
 10  Tipo                   16028 non-null  int64  
 11  NumeroCNPJ             16028 non-null  int64  
 12  RadicalCNPJ            15974 non-null  float64
 13  RazaoSocialRFB         14311 non-null  object 
 14  NomeFantasiaRFB        7470 non-null   object 
 15  CNAEPrincipal          14311 non-null  float64
 16  DescCNAEPrincipal      14241 non-null  object 
 17  Atendida               16028 non-null  object 
 18  CodigoAssunto          16014 non-null  float64
 19  DescricaoAssunto       16014 non-null  object 
 20  CodigoProblema         43 non-null     float64
 21  DescricaoProblema      43 non-null     object 
 22  SexoConsumidor         16022 non-null  object 
 23  FaixaEtariaConsumidor  16028 non-null  object 
 24  CEPConsumidor          13003 non-null  float64
 25  CNPJ                   5654 non-null   float64
 26  Tipo_Pessoa            5654 non-null   object 
dtypes: float64(6), int64(6), object(15)
memory usage: 3.3+ MB
```

In [4]:

```
#Mantendo apenas as colunas que vão interessar
#A única coluna do dataframe da PFN que vamos manter, "Tipo_Pessoa", é a apenas para ir
# Inclusive renomearemos esta única coluna que veio da PFN para 'InscritoDAU'
df = df[['Regiao','UF','Tipo','CNAEPrincipal','Atendida','CodigoAssunto','SexoConsumido
        'CEPConsumidor','Tipo_Pessoa']]
```

In [5]:

```
df.head()
```

Out[5]:

	Regiao	UF	Tipo	CNAEPrincipal	Atendida	CodigoAssunto	SexoConsumidor	FaixaEtariaCo
0	Norte	RO	1	6120501.0	N	187.0	M	entre 51
1	Norte	RO	1	6422100.0	N	53.0	F	entre 41
2	Norte	RO	1	3514000.0	N	185.0	M	entre 41
3	Norte	RO	1	8599604.0	S	236.0	M	entre 31
4	Norte	RO	1	6120501.0	S	187.0	M	entre 51

In [6]:

```
df.shape
```

Out[6]:

```
(16028, 10)
```

In [7]:

```
#Renomeando as colunas
df.rename(columns = {'CNAEPrincipal':'CNAE','CodigoAssunto': 'CodAssunto','FaixaEtariaC
                     'CEPConsumidor':'CEP','Tipo_Pessoa':'InscritoDAU'}, inplace=True)
```

In [8]:

```
1 df.head()
```

Out[8]:

	Regiao	UF	Tipo	CNAE	Atendida	CodAssunto	SexoConsumidor	FaixaEtaria	C
0	Norte	RO	1	6120501.0	N	187.0	M	entre 51 a 60 anos	7682404
1	Norte	RO	1	6422100.0	N	53.0	F	entre 41 a 50 anos	N
2	Norte	RO	1	3514000.0	N	185.0	M	entre 41 a 50 anos	7682432
3	Norte	RO	1	8599604.0	S	236.0	M	entre 31 a 40 anos	7893200
4	Norte	RO	1	6120501.0	S	187.0	M	entre 51 a 60 anos	7893200

In [9]:

```
1 df['InscritoDAU'].value_counts(dropna=False)
```

Out[9]:

```
NaN               10374
Pessoa jurídica    5654
Name: InscritoDAU, dtype: int64
```

Tratamento das colunas "booleanas"

In [10]:

```
1 #A informação que queremos dessa coluna que veio do dataset da PFN é se está inscrito e
2 #Substituímos a presença de algo por "1"; substituímos os "N/A" por zeros, formando, as
3 df['InscritoDAU'] = df['InscritoDAU'].replace('Pessoa jurídica', '1')
4 df['InscritoDAU'].fillna(0, inplace=True)
```

In [11]:

```
1 df['InscritoDAU'].value_counts()
```

Out[11]:

```
0    10374
1     5654
Name: InscritoDAU, dtype: int64
```

In [12]:

```
1 # Fizemos o mesmo para a mais importante coluna do Sindec, a de se a demanda do consumi
2 df['Atendida'] = df['Atendida'].str.replace('S', '1')
```

In [13]:

```
df['Atendida'] = df['Atendida'].str.replace('N', '0')
```

In [14]:

```
df['Atendida'].value_counts()
```

Out[14]:

```
1    9355
0    6673
Name: Atendida, dtype: int64
```

In [15]:

```
# Transformamos em inteiras
df['InscritoDAU'] = pd.to_numeric(df['InscritoDAU'],downcast='integer')
df['Atendida'] = pd.to_numeric(df['Atendida'],downcast='integer')
```

In [16]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16028 entries, 0 to 16027
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Regiao          16028 non-null  object 
 1   UF              16028 non-null  object 
 2   Tipo            16028 non-null  int64  
 3   CNAE            14311 non-null  float64
 4   Atendida        16028 non-null  int8   
 5   CodAssunto      16014 non-null  float64
 6   SexoConsumidor  16022 non-null  object 
 7   FaixaEtaria     16028 non-null  object 
 8   CEP             13003 non-null  float64
 9   InscritoDAU     16028 non-null  int8   
dtypes: float64(3), int64(1), int8(2), object(4)
memory usage: 1.0+ MB
```

Tratamento das outras colunas

In [17]:

```
# 0   Regiao           16028 non-null  object - mantemos pq está sem valores nulos, é
df['Regiao'].value_counts()
```

Out[17]:

```
Sudeste         7470
Norte           3276
Centro-oeste    3070
Nordeste        1403
Sul              809
Name: Regiao, dtype: int64
```

In [18]:

```
#  1   UF    16028 non-null  object mantemos pq está sem valores nulos, é categórica e
df['UF'].value_counts()
```

Out[18]:

```
SP    3363
RO    3217
RJ    2129
GO    2129
MG    1574
RN    1181
MT     849
SC     683
ES     404
CE     113
PB     109
MS      92
RS      75
PA      59
PR      51
Name: UF, dtype: int64
```

In [19]:

```
#2   Tipo              16028 non-null  int64 - retiraremos as que eram "0", pois, segur
# o "Tipo" igual a "O" é de pessoa física
df['Tipo'].value_counts()
```

Out[19]:

```
1    15974
0       54
Name: Tipo, dtype: int64
```

In [20]:

```
# Metodo .loc para filtrar
df = df.loc[(df['Tipo']==1)]
```

In [21]:

```
df['Tipo'].value_counts()
```

Out[21]:

```
1    15974
Name: Tipo, dtype: int64
```

In [22]:

```
#3   CNAE              14311 non-null  float64
# Mudamos para object para não atrapalhar a etapa de ML (é categórica)
# Retiramos os null
df['CNAE'].value_counts()

```

Out[22]:

```
6422100.0    1363
3514000.0    1140
6120501.0     961
6110801.0     841
4753900.0     700
             ...
4520007.0       1
4665600.0       1
3250701.0       1
1811301.0       1
2550102.0       1
Name: CNAE, Length: 407, dtype: int64
```

In [23]:

```
df = df.dropna(subset=['CNAE']).dropna(axis=1, how = 'all')
```

In [24]:

```
df = df.astype({'CNAE': object})
```

In [25]:

```
df.shape
```

Out[25]:

```
(14311, 10)
```

In [26]:

```
#5   CodAssunto        14297 non-null  float64
# Retiramos os null
# Mudamos para object para não atrapalhar a etapa de ML (é categórica)

df = df.dropna(subset=['CodAssunto']).dropna(axis=1, how = 'all')
df = df.astype({'CodAssunto': object})
```

In [27]:

```
df.shape
```

Out[27]:

```
(14297, 10)
```

In [28]:

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 14297 entries, 0 to 16027
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Regiao          14297 non-null  object
 1   UF              14297 non-null  object
 2   Tipo            14297 non-null  int64
 3   CNAE            14297 non-null  object
 4   Atendida        14297 non-null  int8
 5   CodAssunto      14297 non-null  object
 6   SexoConsumidor  14292 non-null  object
 7   FaixaEtaria     14297 non-null  object
 8   CEP             11596 non-null  float64
 9   InscritoDAU     14297 non-null  int8
dtypes: float64(1), int64(1), int8(2), object(6)
memory usage: 1.0+ MB
```

In [29]:

```
1 #6   SexoConsumidor_x  14292 non-null  object
2 # Optamos por retirar as linhas com "N" apenas porque queremos manter as linhas todas p
3 df['SexoConsumidor'].value_counts()
```

Out[29]:

```
F    7422
M    6707
N     163
Name: SexoConsumidor, dtype: int64
```

In [30]:

```
1 #Metodo loc com uso do operador | ("ou") para fazer a query
2 df = df.loc[(df['SexoConsumidor']=='F') | (df['SexoConsumidor']=='M')]
```

In [31]:

```
1 df.shape
```

Out[31]:

```
(14129, 10)
```

In [32]:

```
1 df.head()
```

Out[32]:

	Regiao	UF	Tipo	CNAE	Atendida	CodAssunto	SexoConsumidor	FaixaEtaria	
0	Norte	RO	1	6.1205e+06	0	187	M	entre 51 a 60 anos	768240
1	Norte	RO	1	6.4221e+06	0	53	F	entre 41 a 50 anos	
2	Norte	RO	1	3.514e+06	0	185	M	entre 41 a 50 anos	768243
3	Norte	RO	1	8.5996e+06	1	236	M	entre 31 a 40 anos	789320
4	Norte	RO	1	6.1205e+06	1	187	M	entre 51 a 60 anos	789320

In [33]:

```
1 df['FaixaEtaria'].value_counts()
```

Out[33]:

```
entre 31 a 40 anos    3038
entre 41 a 50 anos    2628
entre 51 a 60 anos    2282
entre 21 a 30 anos    2013
entre 61 a 70 anos    1916
Nao Informada         1132
mais de 70 anos        866
até 20 anos            254
Name: FaixaEtaria, dtype: int64
```

In [34]:

```
1  #7   FaixaEtaria     14297 non-null  object
2 # transformamos em dicionário de categóricos para int
3 dicionario_idade ={'até 20 anos' : 1,
4                    'entre 21 a 30 anos' : 2,
5                    'entre 31 a 40 anos': 3,
6                    'entre 41 a 50 anos': 4,
7                    'entre 51 a 60 anos': 5,
8                    'entre 61 a 70 anos': 6,
9                    'mais de 70 anos': 7
10                  }
```

In [35]:

```
1 df['FaixaEtaria'] = df['FaixaEtaria'].map(dicionario_idade)
```

In [36]:

```
df['FaixaEtaria'].value_counts(dropna = False)
```

Out[36]:

```
3.0    3038
4.0    2628
5.0    2282
2.0    2013
6.0    1916
NaN    1132
7.0     866
1.0     254
Name: FaixaEtaria, dtype: int64
```

In [37]:

```
df['FaixaEtaria'].isna().sum()
```

Out[37]:

```
1132
```

In [38]:

```
df = df.dropna(subset=['FaixaEtaria']).dropna(axis=1, how = 'all')
```

In [39]:

```
df['FaixaEtaria'].isna().sum()
```

Out[39]:

```
0
```

In [40]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 12997 entries, 0 to 16027
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Regiao          12997 non-null  object
 1   UF              12997 non-null  object
 2   Tipo            12997 non-null  int64
 3   CNAE            12997 non-null  object
 4   Atendida        12997 non-null  int8
 5   CodAssunto      12997 non-null  object
 6   SexoConsumidor  12997 non-null  object
 7   FaixaEtaria     12997 non-null  float64
 8   CEP             10519 non-null  float64
 9   InscritoDAU     12997 non-null  int8
dtypes: float64(2), int64(1), int8(2), object(5)
memory usage: 939.2+ KB
```

In [41]:

```
df = df.astype({'FaixaEtaria': int})
```

In [42]:

```
df.shape
```

Out[42]:

```
(12997, 10)
```

In [43]:

```
# 8   CEP             10519 non-null  float64 - retiramos os nulos e trocamos para cate
df = df.dropna(subset=['CEP']).dropna(axis=1, how = 'all')
df = df.astype({'CEP': object})
```

In [44]:

```
df.shape
```

Out[44]:

```
(10519, 10)
```

In [45]:

```
# #Antes de prosseguirmos, promoveremos ajustes no data frame.
# Retiramos a coluna Tipo, pois, conforme acima, ela, agora, tem apenas um valor, "1".
#tratardatasetAED.drop
df.drop(columns=['Tipo'],inplace=True)
```

In [49]:

```
df.shape
```

Out[49]:

```
(10519, 9)
```

In [50]:

```
# Conferindo se ainda há valores nulos
print(df.isna().sum())
```

```
Regiao            0
UF                0
CNAE              0
Atendida          0
CodAssunto        0
SexoConsumidor    0
FaixaEtaria       0
CEP               0
InscritoDAU       0
dtype: int64
```

In [47]:

```
#Exportando para ser trabalhando no notebook de AED
df_aed = pd.read_csv(r'C:\Users\73594253368\Desktop\Curso\Datasets\Procon\df_aed.csv')
```

In [48]:

```
#Exportanto para ser trabalhado no notebook de ML
df.to_csv(r'C:\Users\73594253368\Desktop\Curso\Datasets\Procon\dataset_tratado.csv')
```