FUNDAMENTOS E TÉCNICAS EM CIÊNCIAS DE DADOS

PROF. JOSENALDE OLIVEIRA

josenalde.oliveira@ufrn.br
https://github.com/josenalde/datascience

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - UFRN

PANDAS #4 – CONT. TRATAMENTO DE DADOS

- CORRIGIR VALORES DUPLICADOS

Uso do método duplicated, que retorna Series booleana com ocorrências

```
1 df = pd.DataFrame({'k1': ['um', 'dois']*3 + ['dois'],
2                    'k2': [1,1,2,3,3,4,4]
3                   })
4 df
```

	k1	k2
0	um	1
1	dois	1
2	um	2
3	dois	3
4	um	3
5	dois	4
6	dois	4

```
1 df.duplicated()
```

```
0    False
1    False
2    False
3    False
4    False
5    False
6     True
dtype: bool
```

Neste caso, há um método de tratamento drop_duplicates(), que remove linhas duplicadas

Pode-se também apagar duplicadas com base em colunas específicas

Qual será a saída de drop_duplicates(['k1'])?

PANDAS #4 – CONT. TRATAMENTO DE DADOS

- TRANSFORMAR DADOS COM FUNÇÕES DE MAPEAMENTO

```
1  dict_a = {'alimentos': ['bacon', 'ovo', 'laranja', 'toucinho', 'picanha', 'limão'],
2            'qtd': [10, 1, 1, 20, 30, 20]
3           }
4  df2 = pd.DataFrame(data=dict_a)
5  df2
```

	alimentos	qtd
0	bacon	10
1	ovo	1
2	laranja	1
3	toucinho	20
4	picanha	30
5	limão	20

```
1  dict_alimento_tipo = {
2      'bacon': 'porco',
3      'ovo': 'ave',
4      'laranja': 'fruta',
5      'toucinho': 'porco',
6      'picanha': 'boi',
7      'limão': 'fruta'
8  }
```

O mapeamento é uma espécie de classificação explícita

```
1  df2['tipo'] = df2['alimentos'].map(dict_alimento_tipo)
2  df2
```

	alimentos	qtd	tipo
0	bacon	10	porco
1	ovo	1	ave
2	laranja	1	fruta
3	toucinho	20	porco
4	picanha	30	boi
5	limão	20	fruta

PANDAS #4 – CONT. TRATAMENTO DE DADOS

- DISCRETIZAÇÃO E COMPARTIMENTALIZAÇÃO (BINS): agrupar dados em conjuntos para análise

Exemplo: seja um conjunto de idades. Vamos agrupar nas classes
Jovem: 18,25
Jovem Adulto: 26,35
Adulto: 36, 60
Senior: 61 em diante

- Vamos utilizar o método CUT

```
1  idades = [20,22,25,27,21,23,37,31,61,45,41,32]
```

```
[0, 0, 0, 1, 0, 0, 2, 1, 3, 2, 2, 1]
```

```
1  intervalos = [18,25,35,60,100]
```

Supondo não haver > 100 anos, mas pode ser adaptado...

```
1  categorias = pd.cut(idades,intervalos)
```

```
1  categorias
```

```
[(18, 25], (18, 25], (18, 25], (25, 35], (18, 25], ..., (25, 35], (60, 100], (35, 60], (35, 60], (25, 35]]
Length: 12
Categories (4, interval[int64]): [(18, 25] < (25, 35] < (35, 60] < (60, 100]]
```

Aloca cada dado de idades numa categoria intervalar

.codes/ .categories

```
0: (18, 25],
1: (25, 35],
2: (35, 60],
3: (60, 100]
```

PANDAS #4 – CONT. TRATAMENTO DE DADOS

- DISCRETIZAÇÃO E COMPARTIMENTALIZAÇÃO (BINS): agrupar dados em conjuntos para análise

Exemplo: seja um conjunto de idades. Vamos agrupar nas classes
Jovem: 18,25
Jovem Adulto: 26,35
Adulto: 36, 60
Senior: 61 em diante

- Vamos utilizar o método CUT (ver variação QCUT)

```
1 label_categ = ['jovem', 'jovem adulto', 'adulto', 'senior']
```

```
1 classes = pd.cut(idades, intervalos, labels=label_categ)
```

```
1 classes
```

```
['jovem', 'jovem', 'jovem', 'jovem adulto', 'jovem', ..., 'jovem adulto', 'senior', 'adulto', 'adulto', 'jovem adulto']
Length: 12
Categories (4, object): ['jovem' < 'jovem adulto' < 'adulto' < 'senior']
```

```
1 pd.value_counts(classes)
```

```
jovem          5
jovem adulto   3
adulto         3
senior         1
dtype: int64
```

Exercício: *no dataset de docentes UFRN, criar uma coluna* ***tempo_ufrn*** *com base na coluna* ***data_admissão (ano)*** *e a seguinte classificação: 1980-1990 (pre_internet), 1991-2000 (inicio_internet), 2001-2010 (redes_sociais), 2011-2021 (datascience). Agrupar os docentes com base na nova coluna e elaborar gráfico de barras com o quantitativo de docentes ingressantes nestes 04 períodos*

PANDAS #4 – CONT. TRATAMENTO DE DADOS

- Tratamento de OUTLIERS

Exemplo: eliminar valores fora do intervalo –3 a 3 – truncando

```
#tratamento de outliers
darray = pd.DataFrame(np.random.randn(1000,4))
```

```
darray.describe()
```

	0	1	2	3
count	1000.000000	1000.000000	1000.000000	1000.000000
mean	-0.073627	-0.033275	0.026741	0.018519
std	1.010277	1.032154	1.017073	0.975117
min	-3.062247	-3.237608	-3.031508	-2.913410
25%	-0.758451	-0.735487	-0.672954	-0.615950
50%	-0.058262	-0.016436	0.045986	-0.041536
75%	0.558396	0.692082	0.744549	0.632976
max	3.489405	2.959996	2.949937	3.323118

```
#encontrar na coluna de índice 2 valores com módulo maior que 3
col = darray[2]
```

```
col[np.abs(col) > 3]
```

```
5   -3.031508
Name: 2, dtype: float64
```

```
#linhas com valores maiores que 3 ou -3
darray[(np.abs(darray) > 3).any(1)]
```

	0	1	2	3
5	-3.062247	-1.013566	-3.031508	-0.409121
251	0.773900	-3.237608	0.289440	0.866802
525	0.652107	1.465700	1.128450	3.323118
754	0.134021	-3.197658	-0.269621	-0.441218
865	3.489405	0.242932	1.297426	-0.495738

```
#eliminar valores fora do intervalo -3 a 3:
print(np.abs(darray) > 3)
print(np.sign(darray) * 3)
darray[np.abs(darray) > 3] = np.sign(darray) * 3 # o que for True fica limitado a e 3 ou -3, de acordo com o sinal
darray
```

```
darray.describe()
```

	0	1	2	3
count	1000.000000	1000.000000	1000.000000	1000.000000
mean	-0.074054	-0.032840	0.026773	0.018196
std	1.008484	1.030855	1.016979	0.974074
min	-3.000000	-3.000000	-3.000000	-2.913410
25%	-0.758451	-0.735487	-0.672954	-0.615950
50%	-0.058262	-0.016436	0.045986	-0.041536
75%	0.558396	0.692082	0.744549	0.632976
max	3.000000	2.959996	2.949937	3.000000

PANDAS #4 – CONT. TRATAMENTO DE DADOS

- Tratamento de OUTLIERS

Exemplo: IIQ ou IQR (Intervalo Interquartil)

IQR = Q3 – Q1
lowV = Q1 – 1.5xIQR
highV = Q3 + 1.5xIQR

Se x > highV, x = highV
Se x < lowV, x = lowV