Universidade Presbiteriana Mackenzie

Dataframes: Seleção de Dados

Recursos

Os recursos abaixo podem ser úteis para você aprender e empregar o **Pandas**.

> Pandas

> Pandas Cheat Sheet

Selecionando Sub Conjuntos de Dados no Pandas

Na trilha anterior você aprendeu como obter dados com o Pandas, acessar características básicas de um DataFrame, sumarizar estatísticas dos dados, e selecionar e alterar dados de uma coluna como `pd.Series` do Pandas.

Aqui vamos continuar e aprender como selecionar Sub Conjuntos de Dados de interesse em um DataFrame. Como vimos na trilha anterior, seleções dos dados são bastante importantes pois você nem sempre estará interessado em todos os dados. Por exemplo você pode ter dados de produção de várias unidades de uma fábrica, mas estar interessado somente em dados das unidades de São Paulo (seleção de linhas ou casos). Ou você pode ter dados de vendas com diversas informações dos produtos (cor, modelo etc.) e dos clientes (nome, CPF etc.) e querer apenas dados de peso e dimensões do produto, e da origem e destino da compra para analisar os preços de frete (seleção de colunas ou atributos). Mais frequentemente ainda você vai realizar as duas seleções criando *slices* dos dados.

Seleção de Colunas, Seleção de Linhas e de Linhas e Colunas

Import da biblioteca

```
import pandas as pd
import numpy as np
```

Acessando os Dados

Vamos empregar o mesmo conjunto exemplo de dados da trilha anterior e explorar alguns casos de seleção.

```
df = pd.read_csv('http://meusite.mackenzie.br/rogerio/data_load/tips.csv')

df.head()
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

Selecionando UMA COLUNA (= pd.Series)

Vamos recordar como selecionar uma coluna de dados como uma Série de dados do Pandas. A sintaxe é a seguinte:

```
df.nome_coluna

df [ 'nome_coluna' ]
```

A primeira é mais simples, mas a segunda forma permite a seleção de dados com nomes de colunas com caracteres especiais, brancos ou nomes reservados (**experimente selecionar** `df.size`).

```
valores_de_tip = df.tip
print( type( valores_de_tip ) )
print( valores_de_tip )
```

```
<class 'pandas.core.series.Series'>
0      1.01
1      1.66
2      3.50
3      3.31
4      3.61
       ...
239    5.92
240    2.00
241    2.00
242    1.75
```

```
243    3.00
```

```
print( df.tip.mean(), df.tip.sum(), df.tip.std() )
```

```
2.9982786885245902 731.5799999999999 1.3836381890011826
```

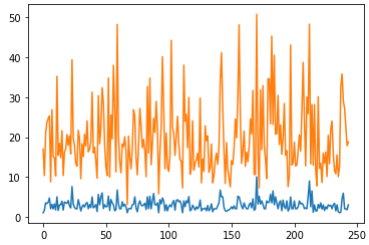
```
df['tip'].mean()
```

```
2.9982786885245902
```

Selecionada uma Série você poderá obter quaisquer informações desse conjunto de dados ou mesmo exibi-lo em um gráfico como veremos nas próximas trilhas.

```
import matplotlib.pyplot as plt

plt.plot(df.tip)
plt.plot(df.total_bill)
plt.show()
```



Selecionando UMA OU MAIS COLUNAS (= pd.DataFrame)

Você pode, entretanto, estar interessado na seleção de mais de uma coluna. Para isso você pode simplesmente informar para o Pandas uma **lista de atributos** a serem selecionados.

```
df [ [ <lista-de-colunas> ] ]
```

Diferentemente da seleção anterior que retorna um Série do Pandas, a seleção desse modo (uma ou mais colunas informadas em uma lista) retorna um DataFrame. **Essa é uma diferença importante que você deve entender**.

```
df.columns
```

```
Index(['total_bill', 'tip', 'sex', 'smoker', 'day', 'time', 'size'], dtype='object')
```

```
df[ ['total_bill', 'tip', 'smoker'] ]   # repare que empregamos uma lista [ ] para indicar a seleção dos atributos
```

	total_bill	tip	smoker
0	16.99	1.01	No
1	10.34	1.66	No
2	21.01	3.50	No
3	23.68	3.31	No
4	24.59	3.61	No
...	...	...	...
239	29.03	5.92	No
240	27.18	2.00	Yes
241	22.67	2.00	Yes
242	17.82	1.75	No
243	18.78	3.00	No

244 rows × 3 columns

```
df[['total_bill', 'tip', 'smoker']].mean()
```

```
total_bill    19.785943
tip            2.998279
dtype: float64
```

Por exemplo, se você estiver interessado em explorar os valores de conta e gorjetas com relação somente sobre o habito de fumar ou não fumar, você pode simplesmente selecionar:

```
tips_fumantes = df[['total_bill', 'tip', 'smoker']]
tips_fumantes.head()
```

	total_bill	tip	smoker
0	16.99	1.01	No
1	10.34	1.66	No
2	21.01	3.50	No
3	23.68	3.31	No
4	24.59	3.61	No

```
type(tips_fumantes)
```

```
pandas.core.frame.DataFrame
```

```
tips_fumantes.describe()
```

	total_bill	tip
count	244.000000	244.000000
mean	19.785943	2.998279
std	8.902412	1.383638
min	3.070000	1.000000
25%	13.347500	2.000000
50%	17.795000	2.900000
75%	24.127500	3.562500
max	50.810000	10.000000

Note que a seleção de **um único atributo a partir de uma lista** produz um DataFrame, enquanto a partir **somente do nome** do produz uma Série do Pandas. Repare a diferença das duas estruturas.

```
tip_DataFrame = df[ ['tip'] ]
print( type(tip_DataFrame) )
print( tip_DataFrame )
```

```
<class 'pandas.core.frame.DataFrame'>
      tip
0    1.01
1    1.66
2    3.50
3    3.31
4    3.61
..    ...
239  5.92
240  2.00
241  2.00
242  1.75
243  3.00

[244 rows x 1 columns]
```

```
tip_Series = df['tip']
print( type(tip_Series) )
print( tip_Series )
```

```
<class 'pandas.core.series.Series'>
0      1.01
1      1.66
2      3.50
3      3.31
4      3.61
       ...
239    5.92
240    2.00
241    2.00
242    1.75
243    3.00
Name: tip, Length: 244, dtype: float64
```

E algumas operações são específicas para DataFrames, não sendo portanto aplicáveis a Séries, enquanto outras são específicas para Séries do Pandas.

```
print( tip_DataFrame.info() )             # info() é um método só para DataFrames
try:
  print( tip_Series.info() )
except AttributeError as err:
  print('Handling run-time error:', err)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   tip     244 non-null    float64
dtypes: float64(1)
memory usage: 2.0 KB
None
Handling run-time error: 'Series' object has no attribute 'info'
```

```
print( tip_Series.to_list() )            # to_list() é um método só para Series
try:
  print( tip_DataFrame.to_list() )
except AttributeError as err:
  print('Handling run-time error:', err)
```

```
Handling run-time error: 'DataFrame' object has no attribute 'to_list'
```

Dica Para um único atributo, de forma geral, empregue a seleção de `pd.Series` quando estiver interessado apenas em selecionar um atributo e a seleção `pd.DataFrame` quando essa seleção será empregada para criação de outros conjuntos de dados (merge de tabelas, novos conjuntos de dados etc.).

Selecionar Linhas

A seleção de linhas é mais interessante por que podemos especificar condições para os valores que buscamos. Por exemplo, você poderia indicar o tipo de peças que deseja ver em um DataFrame com dados de vários componentes, ou indicar a cidade das unidades de fábrica que você tem os dados de produção.

Seu critério de seleção é um predicado lógico e você deve empregar a seguinte sintaxe:

```
df [ <critério de seleção> ]
```

Exemplos

```
df [ nome_coluna == valor ]
df [ nome_coluna != valor ]
df [ nome_coluna > valor ]
```

Assim, podemos selecionar os dados somente dos não fumantes,

```
tips_fumantes[ tips_fumantes.smoker == 'No' ]
```

	total_bill	tip	smoker
0	16.99	1.01	No
1	10.34	1.66	No
2	21.01	3.50	No
3	23.68	3.31	No
4	24.59	3.61	No
...	...	...	...
235	10.07	1.25	No
238	35.83	4.67	No
239	29.03	5.92	No
242	17.82	1.75	No
243	18.78	3.00	No

151 rows × 3 columns

Ou somente dos fumantes,

```
tips_fumantes[ tips_fumantes.smoker == 'Yes' ]
```

	total_bill	tip	smoker
56	38.01	3.00	Yes
58	11.24	1.76	Yes
60	20.29	3.21	Yes
61	13.81	2.00	Yes
62	11.02	1.98	Yes
...	...	...	...
234	15.53	3.00	Yes
236	12.60	1.00	Yes
237	32.83	1.17	Yes
240	27.18	2.00	Yes
241	22.67	2.00	Yes

93 rows × 3 columns

E podemos com isso já responder algumas questões interessantes que envolvem a proporção dos dados de fumantes e não fumantes.

Qual o percental de fumantes nos nossos dados?

```
len( tips_fumantes[ tips_fumantes.smoker == 'Yes' ] ) / len( tips_fumantes )
```

```
0.38114754098360654
```

Seleção de Linhas e Colunas

O uso mais geral é quando fazemos seleções de linhas e colunas dos dados, e às vezes nos referimos a esse Sub Conjunto dos Dados de *Slice* (Fatia dos Dados).

A boa prática indica que sempre faremos primeiro a seleção das linhas,

```
df [ <critério de seleção> ] ...

...Selecionando uma coluna
df [ <critério de seleção> ].nome_coluna                pd.Series
df [ <critério de seleção> ]['nome_coluna']

...Selecionando uma ou mais colunas
df [ <critério de seleção> ][ [<lista-de-colunas>] ]    pd.DataFrame
```

Com isso nossa análise pode fazer muito mais perguntas, que não só sobre o total dos dados e suas proporções.

Fumantes pagam mais gorjeta ou total de contas (em média) que não fumantes?

```
tips_fumantes[ tips_fumantes.smoker == 'Yes' ]['tip'].mean()
```

```
3.008709677419355
```

```
tips_fumantes[ tips_fumantes.smoker == 'No' ]['tip'].mean()
```

```
2.9918543046357624
```

```
tips_fumantes[ tips_fumantes.smoker == 'Yes' ].total_bill.mean()
```

```
20.756344086021507
```

```
tips_fumantes[ tips_fumantes.smoker == 'No' ].total_bill.mean()
```

```
19.18827814569537
```

Exercício. Tente você. Qual a diferença percentual do valor de conta pago por fumantes e não fumantes?

```
tips_fumantes[ tips_fumantes.smoker == 'Yes' ].total_bill.mean() / tips_fumantes[ tips_fumantes.smoker == 'No' ].total_bill.mean() - 1
```

```
0.0817199921962728
```

Exercício. Tente você. Quem em média paga mais gorjetas, homens ou mulheres?

Dica Empregue aqui o DataFrame original, pois tips_fumantes `não tem o atributo` sex` para seleção.

```
df[ df.sex == 'Female' ]['tip'].mean()
```

```
2.833448275862069
```

```
df[ df.sex == 'Male' ]['tip'].mean()
```

```
3.0896178343949052
```

AND, OR, NOT Conditions em Pandas

Para combinar critérios de seleção você pode empregar os operadores lógicos `&` (AND), '|' (OR) ou '!' (NOT), sendo necessário a inclusão de parênteses nas seleções.

```
df[ (df.sex == 'Male') & (df.smoker == 'Yes') ]
```

	total_bill	tip	sex	smoker	day	time	size
56	38.01	3.00	Male	Yes	Sat	Dinner	4
58	11.24	1.76	Male	Yes	Sat	Dinner	2
60	20.29	3.21	Male	Yes	Sat	Dinner	2
61	13.81	2.00	Male	Yes	Sat	Dinner	2
62	11.02	1.98	Male	Yes	Sat	Dinner	2
63	18.29	3.76	Male	Yes	Sat	Dinner	4
69	15.01	2.09	Male	Yes	Sat	Dinner	2
76	17.92	3.08	Male	Yes	Sat	Dinner	2
80	19.44	3.00	Male	Yes	Thur	Lunch	2
83	32.68	5.00	Male	Yes	Thur	Lunch	2
90	28.97	3.00	Male	Yes	Fri	Dinner	2
95	40.17	4.73	Male	Yes	Fri	Dinner	4
96	27.28	4.00	Male	Yes	Fri	Dinner	2
97	12.03	1.50	Male	Yes	Fri	Dinner	2
98	21.01	3.00	Male	Yes	Fri	Dinner	2
105	15.36	1.64	Male	Yes	Sat	Dinner	2
106	20.49	4.06	Male	Yes	Sat	Dinner	2
107	25.21	4.29	Male	Yes	Sat	Dinner	2
138	16.00	2.00	Male	Yes	Thur	Lunch	2
170	50.81	10.00	Male	Yes	Sat	Dinner	3
171	15.81	3.16	Male	Yes	Sat	Dinner	2
172	7.25	5.15	Male	Yes	Sun	Dinner	2
173	31.85	3.18	Male	Yes	Sun	Dinner	2
174	16.82	4.00	Male	Yes	Sun	Dinner	2
175	32.90	3.11	Male	Yes	Sun	Dinner	2
176	17.89	2.00	Male	Yes	Sun	Dinner	2
177	14.48	2.00	Male	Yes	Sun	Dinner	2
179	34.63	3.55	Male	Yes	Sun	Dinner	2

E ainda pode ser útil o operador `.isin()` que verifica a pertinência de um valor a uma lista,

181	23.33	5.65	Male	Yes	Sun	Dinner	2

```
df[ (df.sex == 'Male') & (df.day.isin(['Sat','Sun'])) ]
```

	total_bill	tip	sex	smoker	day	time	size
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
5	25.29	4.71	Male	No	Sun	Dinner	4
6	8.77	2.00	Male	No	Sun	Dinner	2
...	...	...	...	...	...	...	...
236	12.60	1.00	Male	Yes	Sat	Dinner	2
237	32.83	1.17	Male	Yes	Sat	Dinner	2
239	29.03	5.92	Male	No	Sat	Dinner	3
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2

117 rows × 7 columns

Exercícios

Q0. Considere nossa base exemplo Tips. Considerando somente os finais de semana, quem tem média de participantes na mesa maior, fumantes ou não fumantes?

```
df = pd.read_csv('http://meusite.mackenzie.br/rogerio/data_load/tips.csv')

df[ (df.smoker == 'Yes') & (df.day.isin(['Sat','Sun'])) ]['size'].mean()
```

```
2.5081967213114753
```

```
df[ (df.smoker == 'No') & (df.day.isin(['Sat','Sun'])) ]['size'].mean()
```

```
2.764705882352941
```

Exercício: **CASE Insurance**

Acesse a base de dados http://meusite.mackenzie.br/rogerio/data_load/insurance.csv para as questões a seguir.

```
df = pd.read_csv('http://meusite.mackenzie.br/rogerio/data_load/insurance.csv')
df.head()
```

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200

Q1. Qual percentual de homens e mulheres assegurados na base?

```
fem = len( df[df.sex == 'female'] ) / len(df)
print(f'Percentual de Mulheres seguradas: {fem:.2f}')

masc = len( df[df.sex == 'male'] ) / len(df)
print(f'Percentual de Homens segurados: {masc:.2f}')
```

```
Percentual de Mulheres seguradas: 0.49
Percentual de Homens segurados: 0.51
```

Q2. Qual o total dos valor de seguros pago por homens e mulheres segurados na base?

```
fem = sum( df[df.sex == 'female'].charges )
print(f'Total dos Seguros (Mulheres):  {fem:.2f}')

masc = sum( df[df.sex == 'male'].charges )
print(f'Total dos Seguros (Homens):  {masc:.2f}')
```

```
Total dos Seguros (Mulheres):  8321061.19
Total dos Seguros (Homens):  9434763.80
```

Q3. Qual o precentual de valor de seguros pagos por homens e mulheres segurados na base?

```
fem = sum( df[df.sex == 'female'].charges ) / sum( df.charges )
print(f'Total dos Seguros (Mulheres):  {fem:.2f}')

masc = sum( df[df.sex == 'male'].charges ) / sum( df.charges )
print(f'Total dos Seguros (Homens):  {masc:.2f}')
```

```
Total dos Seguros (Mulheres):  0.47
Total dos Seguros (Homens):  0.53
```

Q4. Não fumantes apresentam em média mais filhos?

```
df[df.smoker == 'yes'].children.mean() > df[df.smoker == 'no'].children.mean()
```

```
True
```

Q5. Qual a média e maior de idade dentre os segurados homem, das regiões northwest ou northeast?

```
df[ (df.sex == 'male') & (df.region.isin(['northwest','northeast']) ) ].age.mean()
```

```
38.848765432098766
```

```
df[ (df.sex == 'male') & (df.region.isin(['northwest','northeast']) ) ].age.max()
```

```
64
```

Exercício: **CASE: European Energy:**

Qual o tipo de Energia mais produzido na Europa? O uso de fontes renováveis já é significativo?

European Energy link

Explore aqui como é a produção de Energia na Europa empregando a base de dados abaixo. Você pode achar útil fazer uma exploração inicial dos dados antes de começar.

```
import pandas as pd
df = pd.read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2020/2020-08-04/energy_types.csv')

df.head()
```

	country	country_name	type	level	2016	2017	2018
0	BE	Belgium	Conventional thermal	Level 1	30728.0	31316.0	30092.635
1	BE	Belgium	Nuclear	Level 1	41430.0	40128.5	26995.628
2	BE	Belgium	Hydro	Level 1	1476.0	1360.9	1239.248
3	BE	Belgium	Pumped hydro power	Level 2	1110.0	1093.2	983.190
4	BE	Belgium	Wind	Level 1	5340.0	6387.9	7177.346

Q6. Quantos tipos de energia e países diferentes há na base?

```
df.type.unique()
```

```
array(['Conventional thermal', 'Nuclear', 'Hydro', 'Pumped hydro power',
       'Wind', 'Solar', 'Geothermal', 'Other'], dtype=object)
```

```
len( df.country_name.unique() )
```

```
37
```

```
# cuidado, não use
# df.country_name.count()
```

Q7. Qual o percentual de energia nuclear produzido na Europa no ano de 2018?

```
df[df.type == 'Nuclear']['2018'].sum() / df['2018'].sum()
```

```
0.22925252769137672
```

Q8. Qual o percentual de energia nuclear produzido na França no ano de 2018?

```
df[ (df.type == 'Nuclear') & (df.country_name == 'France')]['2018'].sum() / df[(df.country_name == 'France')]['2018'].sum()
```

```
0.7010980377654339
```

Q9. Qual o percentual de energias limpas produzida na Europa no ano de 2018?

Energias limpas somente Hidroelétrica, Eólica, Solar e Geotérmica.

```
df[ df.type.isin(['Hydro',
                  'Wind',
                  'Solar',
                  'Geothermal']) ]['2018'].sum() / df['2018'].sum()
```

```
0.3053953330199326
```

Q10. Considerando os anos de 2016, 2017 e 2018, a produção de energias limpas tem aumentado na Europa? Verifique em termos absolutos e percentuais.

Energias limpas somente Hidroelétrica, Eólica, Solar e Geotérmica.

```
# Você pode fazer separadamente, mas se empregar os recursos de programação
# que aprendeu até aqui, como o for, listas etc. ficará bem mais inteligente...

clean_energy = ['Hydro', 'Wind', 'Solar', 'Geothermal']
years = ['2016','2017','2018']

for year in years:
  print('Total do ano ', year, '\n', df[ df.type.isin(clean_energy) ][year].sum() )
  print('% do ano ', year, '\n', df[ df.type.isin(clean_energy) ][year].sum() / df[year].sum() )
```

```
Total do ano  2016
 1072303.537
% do ano  2016
 0.28338767596296544
Total do ano  2017
 1077515.942
% do ano  2017
 0.281950926960757
Total do ano  2018
 1156738.155
% do ano  2018
 0.3053953330199326
```