

# Introdução à Análise de Dados com o Pandas

Posted by Anderson Berg (<https://andersonberg.github.io/pythonize-pelican/author/anderson-berg.html>) on 08 Ago, 2017

Pandas (Python Data Analysis) é uma biblioteca de alto desempenho para análise de dados. O pandas tem diversas ferramentas para tratamento e preparação de dados. Combinando com o IPython também tem funcionalidades para análise e modelagem de dados. Neste artigo vou mostrar através de exemplos bem práticos como utilizar o pandas e IPython para fazer tratamento e análise de um grande volume de dados.

Pra iniciar, vou mostrar como fazer o tratamento de dados a partir de um arquivo excel. O exemplo que vou utilizar é do resultado do censo no estado de Pernambuco. Você pode baixar os arquivos excel no site do [IBGE](http://www.ibge.gov.br/home/estatistica/populacao/censo2010/resultados_gerais_amostra/resultados_gerais_amostra_tab_xls.shtm) ([http://www.ibge.gov.br/home/estatistica/populacao/censo2010/resultados\\_gerais\\_amostra/resultados\\_gerais\\_amostra\\_tab\\_xls.shtm](http://www.ibge.gov.br/home/estatistica/populacao/censo2010/resultados_gerais_amostra/resultados_gerais_amostra_tab_xls.shtm))

Carregar dados de um arquivo excel é bem simples com a função "read\_excel". Você pode exibir parte da tabela com o ".head()", que vai mostrar as 5 primeiras linhas:

In [1]:

```
import pandas as pd
df_pop = pd.read_excel("total_populacao_pernambuco.xls")
df_pop.head()
```

Out[1]:

	Código do município	Nome do município	Total da população 2000	Total de homens	Total de mulheres	Total da população urbana	Total da população rural	Total da população 2010
0	2600054	Abreu e Lima	89039.0	45165.0	49263.0	86589.0	7839.0	94428.0
1	2600104	Afogados da Ingazeira	32922.0	16790.0	18301.0	27406.0	7685.0	35091.0
2	2600203	Afrânio	15014.0	8751.0	8837.0	5859.0	11729.0	17588.0
3	2600302	Agrestina	20036.0	10938.0	11742.0	16955.0	5725.0	22680.0
4	2600401	Água Preta	28531.0	16581.0	16465.0	18708.0	14338.0	33046.0

No exemplo, "df\_pop" é uma estrutura de dados chamada DataFrame. O DataFrame tem duas dimensões e transforma os dados em uma tabela. Cada linha ou coluna de um DataFrame possui outra estrutura do pandas chamado de Series, que nada mais é do que um array unidimensional.

Suponha agora que você quer remover algumas colunas desnecessárias para melhorar a visualização e facilitar a manipulação dos dados. Trabalho simples de ser feito no pandas:

In [2]:

```
df_pop = df_pop.drop('Código do município', axis=1)
```

Para remover várias colunas ao mesmo tempo, é um pouco diferente:

In [3]:

```
df_pop = df_pop.drop(df_pop.columns[[2, 3, 4, 5]], axis=1)
df_pop.head()
```

Out[3]:

	Nome do município	Total da população 2000	Total da população 2010
0	Abreu e Lima	89039.0	94428.0
1	Afogados da Ingazeira	32922.0	35091.0
2	Afrânio	15014.0	17588.0
3	Agrestina	20036.0	22680.0
4	Água Preta	28531.0	33046.0

A função "drop" retorna um novo DataFrame sem as colunas especificadas, por isso temos que atribuir novamente ao DataFrame original. Para evitar isso, você pode utilizar o atributo "inplace".

Vamos renomear algumas colunas e utilizar o inplace para modificar diretamente o DataFrame original:

In [4]:

```
df_pop.rename(columns={"Total da população 2000": "Total 2000", "Total da população 2010": "Total 2010"}, inplace=True)
df_pop.head()
```

Out[4]:

	Nome do município	Total 2000	Total 2010
0	Abreu e Lima	89039.0	94428.0
1	Afogados da Ingazeira	32922.0	35091.0
2	Afrânio	15014.0	17588.0
3	Agrestina	20036.0	22680.0
4	Água Preta	28531.0	33046.0

Dá uma olhada agora no final da tabela, tem alguns valores vazios ali

In [5]:

```
df_pop.tail()
```

Out[5]:

	Nome do município	Total 2000	Total 2010
182	Vicência	28820.0	30731.0
183	Vitória de Santo Antão	117609.0	130540.0
184	Xexéu	13597.0	14092.0
185	Pernambuco	7918344.0	8796032.0
186	NaN	NaN	NaN

Problema simples de resolver com o pandas. Você tem duas opções aqui:

1. Preencher com um valor padrão
2. Remover as células vazias

Se tudo que você precisa é preencher as lacunas, use a função "fillna":

In [6]:

```
filled_df = df_pop.fillna(axis=0, value=0)
filled_df.tail()
```

Out[6]:

	Nome do município	Total 2000	Total 2010
182	Vicência	28820.0	30731.0
183	Vitória de Santo Antão	117609.0	130540.0
184	Xexéu	13597.0	14092.0
185	Pernambuco	7918344.0	8796032.0
186	0	0.0	0.0

Por outro lado, se a melhor decisão é remover essa linha:

In [7]:

```
df_pop.dropna(axis=0, how='all', inplace=True)
df_pop.tail()
```

Out[7]:

	Nome do município	Total 2000	Total 2010
181	Vertentes	14957.0	18267.0
182	Vicência	28820.0	30731.0
183	Vitória de Santo Antão	117609.0	130540.0
184	Xexéu	13597.0	14092.0
185	Pernambuco	7918344.0	8796032.0

Uma última função que eu queria comentar neste post é a função map. Ela é bem útil se você precisa modificar todos os elementos de uma linha ou coluna. No nosso exemplo podemos converter todos os números de float para int:

In [8]:

```
df_pop["Total 2000"] = df_pop["Total 2000"].map(lambda x: int(x))
df_pop["Total 2010"] = df_pop["Total 2010"].map(lambda x: int(x))
df_pop.head()
```

Out[8]:

	Nome do município	Total 2000	Total 2010
0	Abreu e Lima	89039	94428
1	Afogados da Ingazeira	32922	35091
2	Afrânio	15014	17588
3	Agrestina	20036	22680
4	Água Preta	28531	33046

Basicamente, a função map recebe como parâmetro outra função que será mapeada para os elementos de uma coluna ou linha.

Queria ser breve neste post, mostrando algumas operações básicas que podem ser feitas com o pandas. Em um próximo post vou trazer mais alguns detalhes sobre o pandas e como fazer operações mais avançadas com análise de dados.

## Comments !

0 Comentários

pythonize.org

1 Entrar

Recomendar

Compartilhar

Ordenar por Mais votados



FAZER LOGIN COM


OU REGISTRE-SE NO DISQUS

Seja o primeiro a comentar.

TAMBÉM EM PYTHONIZE.ORG

### 20 artigos e sites essenciais para você aprender Python


5 comentários • 7 meses atrás



Bruno Rocha — Flask -> <http://bit.ly/whattheflask>

### Transformando seu Código Python em Executável com o py2exe


3 comentários • 6 meses atrás



Emerson Lara — Muito bom, camarada. Funciona mesmo. Porém, no Python 3.x o py2exe não funciona. Tem que usar o CX\_FREEZE. Tem as ...

### Python - Dividindo uma Lista em N Partes


1 comentário • 6 meses atrás



maurobaraldi — Legal a solução proposta, mas se tiver um objeto com muitos itens, pode ter um problema de performance com as listas. Como ...

### Como Recuperar Posts do Twitter Rapidamente com Python

13 comentários • 6 meses atrás



Nádía Félix — Oi Anderson, este método GetPublicTimeline() ainda existe na Api? Estou tentando usar e está retornando um erro. Estou ...

Inscreva-se

Adicione o Disqus no seu siteAdicionar DisqusAdicionar

Privacidade



Blog powered by [Pelican \(http://getpelican.com\)](http://getpelican.com), which takes great advantage of [Python \(http://python.org\)](http://python.org).

